I/O NEWS

Soft Tools
Software Resource Guide
New Products

THE OFFICIAL PUBLICATION OF THE INTERNATIONAL ASSOCIATION OF CROMEMCO USERS

Volume Five, Number Five

Single Copy Price $10.00

SYSTEM 120 Announced by Cromemco

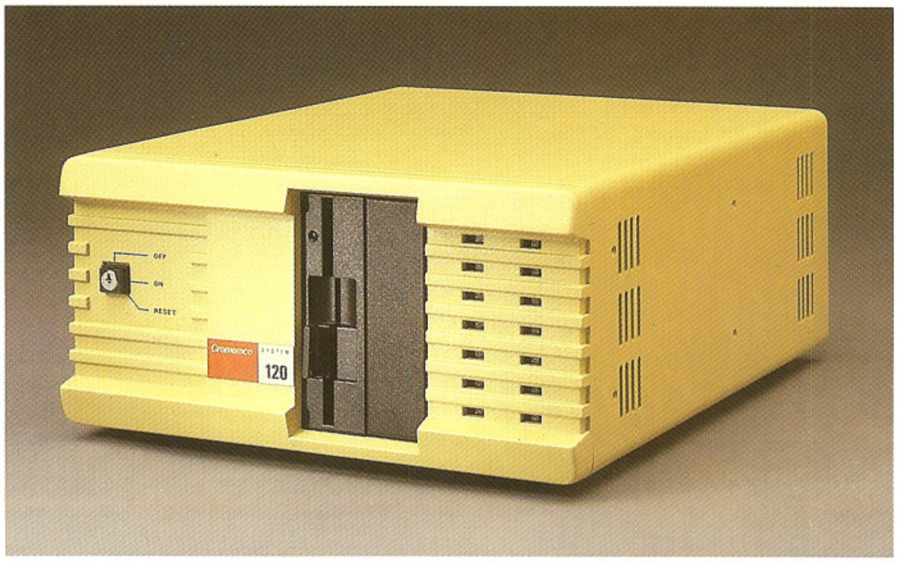
Following closely on the heels of the highly successful System 420, Cromemco has introduced the new System 120 computer. Like the CS420, the CS120 uses Cromemco's new 32-bit XXU processor card with on-board math coprocessor. While the CS420 is based on Cromemco's 21-slot CS400 chassis though, the CS120 is based on the smaller, 8-slot, CS100 chassis.

Even in its smallest configuration the CS120 includes 2 megabytes of RAM memory. The basic card complement for the computer includes the XXU processor, 64FDC floppy disk controller, STDX hard disk controller, 2048KZ memory, and Octart 8-channel serial interface. This configuration includes the Cromix Plus operating system and is available with either a 50 megabyte or 140 megabyte internal hard disk drive. Three slots are available for system expansion with additional 2048KZ memory cards or Octart serial interfaces.

The basic system with the UNIX System V.2 operating system also includes the XMU memory management card, leaving 2 slots available for system expansion. While the UNIX-based system is also available with either 50 megabyte or 140 megabyte hard disk, Cromemco recommends the 140 megabyte drive for all but dedicated UNIX applications, due to the large size of the UNIX operating system and the on-line documentation.

By far the most exciting thing about the CS120 is its **performance**. Users of

Continued on page 7



MAXIMIZER Benchmark Tests for Single and Multi-user Environments

by Dr. Stephen Huber

In a recent issue of *I/O NEWS (Vol. IV, No. 6)* the performance characteristics of the MAXIMIZER Co-processor set were discussed with some attention given to Whetstone Benchmark results. Although that article was an excellent introduction to the characteristics of the MAXIMIZER for the potential user, clearly a more comprehensive and realistic set of test results were needed in order to determine if the MAXIMIZER would be appropriate to a particular user's needs. For example, as a theoretical physicist I use my Cromemco System III primarily for number crunching activities and am in need of substantial speed improvement.

This article addresses two areas of testing.

Continued on page 31

How to Write Software for PEOPLE

by Dr. Jeff Johnson

Not long ago, a friend gave me a disk containing some games that would run on my C-10. One of them was an adventure game. I fired it up and began typing in adventure-like commands (e.g., "go north", "take fish", etc.), but it responded to everything with "I don't understand that." Though that program undoubtedly took some poor computer nerd many weeks to write, it took me only ten seconds to delete it from my disk. I have no patience with that kind of nonsense.

A few years ago, while walking around the exhibition floor at a West Coast Computer Faire, a colleague and

Continued on page 29

IT'S HERE!! dBIII Compiler for UniPlus+ System V

Software Standards is now shipping dBIIICompiler from WordTech Systems for Cromemco's UniPlus System V machines. dBIIICompiler is a powerful dBase III compiler as well as a full-function program development system. No other software or product (such as dBase III) is required to use dBIIICompiler although it is compatible with dBase II and dBase III files from any other machine. With dBIIICompiler under UniPlus System V, you can quickly develop or port anything from small utilities to giant multi-user data processing systems. In addition to all the features found in our Cromix-Plus version, including multi-user record level locking, terminal independence and dBase II data file compatability, our new UniPlus version generates memory-efficient **SHARED CODE COMPILED PROGRAMS** and is 100% source code compatible with our Cromix-Plus version. Our latest release also incorporates **MULTIPLE PRINTER** support and complete control over the spooler. Suggested list price is $995.00 and there are no royalties for distributing compiler programs.

RealWorld Accounting Packages

RealWorld is a full-featured accounting system which includes A/P, A/R, G/L, Payroll, Inventory, Order Entry, Billing, and Sales Analysis. All modules are fully integrated and the system is multi-RealWorld is available on both Z80 and 68000 Cromix as well as UniPlus System V. Suggested list price is $695.00 for Z80 versions and $795.00 for 68000 Cromix and UniPlus Versions.

Contact your Cromemco dealer for more information.
Dealers contact Software Standards for Discount Schedules, Literature and Demo Packages.

Software Standards, Inc.
WE'VE MOVED!!!
P.O. Box 3181
Lake Charles, La 70602-3181
Phone (318) 433-3690

UniPlus is a trademark of UniSoft Systems
dBase III is a registered trademark of Ashton-Tate
Cromix-Plus is a trademark Cromemco, Inc.
dBIIICompiler is a trademark of WordTech Systems, Inc.
RealWorld is a trademark of RealWorld Corp.

table of contents

November/December, 1986 Volume Five, Number Five

I/O NEWS *The Official Publication of The International Association of Cromemco Users is available through membership in the association. Editorial and advertising policies are designed for the enlightenment of the members in regard to new uses for, and developments of, Cromemco products and other products compatible with Cromemco systems.*

I/O NEWS [ISSN 0274-9998] is published bi-monthly by The International Association of Cromemco Users (a California corporation), P.O. Box 17658, Irvine, CA 92713. General Offices are at 34021 Granada #B, Dana Point, CA 92629. Telephone: (714) 661-9764. Second Class Postage Paid at Dana Point and Santa Ana, CA. **POSTMASTER: Send address changes to I/O NEWS, P.O. Box 17658, Irvine, CA 92713.**

Subscriptions to I/O NEWS are entered with membership in The IACU. Yearly memberships may be purchased for $42 (U.S. delivery address), $49 (delivery address in Canada or Mexico), and $60 (other international delivery addresses). Contact IACU for multi-year membership rates. Back issues of I/O NEWS are available for $10.00 per issue. Please note: all prices are in U.S. dollars.

Return postage must be included with all manuscripts and photos submitted if they are to be returned. The IACU and I/O NEWS accept no responsibility for the return of unsolicited materials.

All rights in letters sent to IACU and I/O NEWS will be treated as unconditionally assigned for publication and copyright to comment editorially and to edit.

Copyright © 1986 by The International Association of Cromemco Users. All rights reserved. Nothing may be reprinted in whole or in part without written permission of the publisher.

William E. Jaenicke
Editor and Publisher

Lisa B. Jaenicke
Business Manager

Art Direction — Howard Millman
Printing — Hallmark Litho
Typography — Western Outdoors

COVER FEATURES

ARTICLES & FEATURES

DEPARTMENTS

INPUT...

Editor:

I had a System Three with ZPU, Tuart, 64FDC, 256KZ, WDI-II and a HD-20. I decided to convert it to a CS-300 so I bought a STDC, OCTART, 1024KZ, XPU, and a kit to convert my hard disk from 20 Mb to 21 Mb. I installed the Cromix-Plus 31.05 and the sbasic68 and passed all of my software to the new Basic. Everything works like I expected except that I have troubles with the expand instruction. For example:

```
10 Dim A$(4)
25 A$ = "613"
30 Do While Len(A$) < 5
35 Expand A$(0),1: A$(0,0) = "0"
40 Endwhile
45 Print A$
50 End
```

In this case the program falls into an infinite loop and I have to interrupt the execution with Esc. It doesn't do the same with the 32K Sbasic or the Sbasic.bin (from Systems Atlanta). I had to modify all of the software where this instruction was used and then I could work without troubles, but it took a few days to make the changes. My question is: Is it right or is my version of Basic bad?

The other thing is that I wanted to add the 256KZ to my system and have 5 banks of 256Kb (I modified the sysdef) but when I boot my system the computer doesn't recognize the 5th bank of 256K. What can I do?

My version of Basic is: STB-DL. Release 1.

My 256KZ has the numbers: 256 KZ-020-0125. Rev. A1. Mod level 2.

Thanks for all your time and your help and receive my regards.

Ing. Juan Ramon Lopez Ruiz
Yucatan, Mexico

Editor's Reply:

I verified the problem with 68000 Sbasic expand function. The program you supplied would indeed go into an infinite loop, although it is syntactically correct. By using the trace option it became evident that A$ was being treated by expand as having a length of four (4) instead of five (5) as specified in the DIM statement. On each pass it would indeed expand the string, inserting a "0" at the front. However, after it filled the string to length 4 it would shove subsequent characters off the right-hand side of the string. Thus, it never attained a length of 5 and looped continuously.

The solution is to dimension A$ as five, i.e., Dim A$(5). Then the program works as expected. I admit that this is not an elegant solution, and would still necessitate program modifications. Nevertheless, these types of changes may be less than writing your own expand function or subroutine.

The problem you experienced in attempting to add another 256KZ board to your system could result from a number of things. For one, verify that the address switch settings on all of the boards are correct. If that is not the problem, verify that the board is working by swapping places with another board, i.e., change its position in the bus. Finally, try re-addressing the board to occupy a different bank of memory. By rebooting, RDOS will perform its memory check and determine whether or not the board is functioning properly.

Editor:

In the *Volume V, No. 3* edition of *I/O NEWS*, Dr. Beer in *C-10 ENCOUNTERS* challenged readers to "describe any way of returning from CDOS to CROS without turning off the machine." The answer is in Cromemco's Application Note 023-9102, *C-10 I/O Port Assignments*, dated 04/18/83. In brief, outputting a 1 to port 40h switches in ROM in address space 4000h to BFFFh, which includes CROS starting at address 8000h. From there, the solution is simple. The following three Z80 assembly instructions do the job:

```
LD A,1
OUT 40H,A
JP 8000H
```

In hexadecimal code that translates to:

```
3E 01 D3 40 C3 00 80
```

This code must reside below 4000h in memory. It can be poked out with BASIC and then executed as a subroutine (from which control will not return), or written as a .COM file, either with DEBUG or the assembler and linker or loader. I used DEBUG to create the code and the file SYSRESET.COM containing it. Note: Do NOT single step through this code in DEBUG! You'll hang the system!

When this code is executed, the power-on sequence of the machine is invoked, the screen is blanked, the bell rings, and after a pause while memory is tested, the system goes through the normal boot load sequence, looking for a disk and trying to boot CDOS if it finds one. But as soon as you hear the bell, press ESCAPE, and presto!, you're in CROS, with the ';' prompt. From there you can do anything.

Since we're talking about the C-10 and CROS, let me describe a peculiarity of the system. Cromemco's documentation says that the C-10 cannot read single-density diskettes. In fact, it can, though it cannot format them. It can write them too, but here's the catch. If you write to a single-density diskette from a location memory that lies over CROS, the system will switch in ROM during the write, and what is written will be the CROS code itself, instead of the contents of RAM memory. As soon as the write completes, RAM is switched back in. Nothing is lost in memory, but you can't get it to the disk. This is transparent to all programs, including DEBUG. I would guess that CDOS is using hardware assistance by calling on CROS to do the job, and that is consequence. Not all of ROM is switched in, or at least it is only switched in of the data overlays CROS beginning at 8000h, so it's not quite that simple. (This happens on machines that don't have Release 5 installed, with CDOS itself in ROM. I don't know if Release 5 changes anything).

Gerald Reynolds
Helderberg College
Republic of South Africa

Editor's Note:

The following is an excerpt from a letter, received from Ben Ross Schneider, a long-time Cromemco user and IACU member who recently adopted a PC clone ...

... So I am afraid that Cromemco is on the way out unless they admit defeat and use their great creativity and knowhow to make better IBM clones, and possibly moving in the direction of Amiga and Apple with their 68000 machines. I was sad to see George Morrow throw in the towel, even after going the IBM-compatible route. And to think that the Z80 could have grown in speed and memory access just as the 8088 has! If it weren't for IBM's cynical and ruthless establishment of the 8088 with the pre-meditated intent of murdering the Z80, I would be experiencing the speed, capacity, and low prices on an upgrade of my Z-2D that I now have with a PC clone.

The good news is that the Big Blue mystique is gone. They have created a new standard which they themselves can't change. The sucker market is gone. PCs are taking over the work of mainframes. IBM makes most of its money on mainframes. You heard it first from me: IBM has reached the peak of its development and will decline from now on. Its whole success is based on its name, not its products. But now Big Brother Blue is wearing no clothes. The audience is now looking at the magician's other hand.

The book just published by Richard Thomas DeLamarter (Big Blue: IBM's Use and Abuse of Power, Dodd, Mead, November 10th) will certainly contribute to the disrobing. As senior economist of the US Justice Department in the preparation of the discontinued anti-trust suit against IBM he is an unquestionable authority. He has done an immense amount of homework and gives us a 398-page brief against IBM. His verdict is: Guilty of anti-trust violation by virtue of idiosyncratic and malicious product-design intended to make users IBM-dependent and keep them so.

What they have done to Cromemco is exactly what they have done to all would-be competitors in all fields of computer activity. Theirs was an artificial eminence, and in my opinion they can no longer sustain the image that propped them up.

So we can take comfort in that, at least.

Ben Ross Schneider, Jr.
Lawrence University
Department of English
Appleton, WI 54912

Editor's Reply:

... Your comment that "Cromemco is on the way out unless they admit defeat and use their great creativity and knowhow to make better IBM clones ..." raised my eyebrows. I don't think Cromemco has ever been too concerned about IBM. They don't see themselves as being in competition with IBM for theirs is not the personal computer market.

As early as 1980, as evidenced by the announcement of the multi-user, multi-tasking Cromix Operating System on the front cover of the premier issue of *I/O NEWS*, Cromemco was moving away from single-user "personal computers." They were concerned with penetrating markets in science & engineering with their state of the art hardware and software development tools.

In short, the "business market" (which became dominated by IBM and spawned legions of clones) has never been a primary target for Cromemco computers. That is not to say that Cromemco computers are not in use running business applications. They are — a great many of them. The development of business application software under the Cromemco development languages and operating systems did occur. Some was absorbed from CP/M, for through some degree of foresight Cromemco made CDOS, and by way of the CDOS Simulator, Cromix, capable of running a good deal of applications software developed under CP/M (dBASE, WordStar, etc.) But it was nothing like the proliferation of software under MS-DOS. As you know, all of that changed when the PC came on the scene. And from what you have stated, the downfall of the Z-80 and CP/M was engineered by Big Blue. But by this time Cromemco was well into their own multi-user, multi-tasking operating system — Cromix. And aside from testing the PC waters with their C-10 (which was too little, too late) they had set their sights away from the PC and towards the minicomputer and mainframe computer with systems that were now being called "supermicrocomputers."

Today, with their state-of-the-art 68000- based line of UNIX systems, it is an exercise in futility to compare a Cromemco System (of whatever flavor) with an IBM PC (of whatever flavor). Cromemco will never develop the often mentioned Cromemco/IBM compatibility bridge (although some enterprising individual or group might). In Cromemco's scheme of things the PC/clone is relegated to being a terminal or stand-alone workstation connected to a Cromemco multi-user system, and communicating by way of any of a number of available communication packages. Most people believe that UNIX will become the standard operating system, and UNIX software development is expanding at an enormous rate. Cromemco recognized this and offers a very standard port of UNIX System V. They also have an in-house department responsible for 3rd party software acquisition. Part of the work that I do here is directed to that end. After all, it was the vast selection of software that made the PC so appealing. If UNIX does become standard then the question of software availability will become insignificant; the primary factor will be the fitness of the hardware, and that is an area where Cromemco beats IBM hands down. You pointed out that IBM had hurt itself when its own micros began to encroach upon the mainframe domain. If that is the case, then Cromemco has been hurting them all along. In the future they may do it in even a bigger way. And that, too, is something we can take comfort in.

FOR SALE

System III (purchased 1980) with 16FDC, 2 8-inch DD Persci, Tuart, 64K Ram.

3102 Terminal and Keyboard.

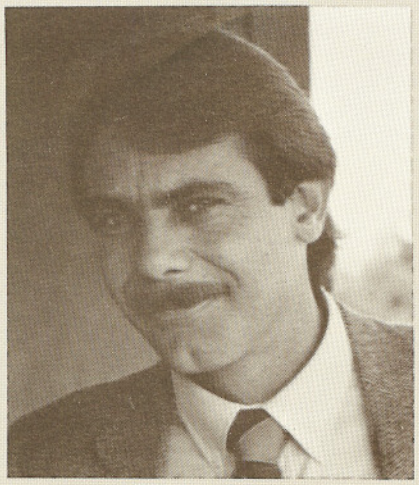
3355A Printer (NEC printer with Cromemco name on it — presently not working).

Plus software, documentation (CDOS, Structured Basic, RATFOR, Wordprocessing Formatter) and manuals. $980.

Quietizer 9821 printer cover ($300 value) for $200.

Contact Valerie Berry
(916) 972-1137

OUTPUT...


Bill Jaenicke

Lisa Jaenicke

Happy New Year!

By the time you read this the New Year will be upon us. So I will take the time now to wish all of you the best for 1987. What with the flurry of new and exciting developments I'm sure that it will be an exhilarating new year for all of us!

And with 1986 behind us, I want to personally thank the column editors, editorial contributors, the folks at Cromemco, our faithful advertisers, Howard Millman (our art director), my wife Lisa, and all the rest of you who have had a part in making each issue of *I/O NEWS* something special. With similar efforts this year the IACU and *I/O NEWS* are sure to prosper.

Software Resource Guide

One of 1986's projects was the publication of the Software Resource Guide (SRG) for Cromemco Systems. Without a doubt, something of this nature is needed, but many of you voiced a legitimate concern over how updates would be handled, the cost of these updates, their frequency, etc. In short, the format in which the guide was presented obviously was not acceptable.

And so we live and we learn. Rather than abandon the idea, for it is surely something which could be of great benefit to all, we look at other alternatives. 1987 will see something new in the way of a Software Resource Guide. The eventual form it will take will depend largely on your suggestions. What is needed is a format in which additions and changes can be readily made, and to be able to do so in such a way that the costs of updating and distributing the information does not become prohibitive.

In our case, if we were to publish new printed volumes, they would be few and far between because of the high cost of printing and mailing. The same would be true even if the format were such that printed updates could be issued and inserted into a binder containing an existing guide. This would require an updated table of contents and indexes and a peculiar method of numbering pages, all of which means a whole lot of work on someone's part (mine). There has to be a better way. Our computers can provide the way. They afford us the most economical means of distributing this type of information in formats specific to one's individual requirements. For instance, the SRG could be made available in a database format (dBASE, Informix, Unify); it could be provided in an ASCII format (SDF) which could be read into one's own particular database; or it might take the form of an ASCII text file which could be printed or for which SCREEN or CE could be used to "FIND" specified character sequences (types of programs, keywords, etc.). Best of all, it could be set up on the IACU system and accessed via modem.

Perhaps all of these options must be employed: what your particular requirements are will depend on your system — C-10, S-100 CDOS, Cromix, Cromix-Plus, or UNIX System V — with modem or without. In any case, something along these lines will be available for IACU members early in 1987.

Where Have all the Gadgets Gone?

Another area that seems to demand some attention, and which I hope will get more coverage, is the realm of gadgetry. Where are the hobbyists and entrepreneurs that create the boards that enable our systems to do some of the menial (and not so menial) tasks that think that the Z80 could have grown in speed and memory access just as the 8088 has! If it weren't for IBM's cynical and ruthless establishment of the 8088 with the pre-meditated intent of murdering the Z80, I would be experiencing the speed, capacity, and low prices on an upgrade of my Z-2D that I now have with a PC clone.

The good news is that the Big Blue mystique is gone. They have created a new standard which they themselves can't change. The sucker market is gone. PCs are taking over the work of mainframes. IBM makes most of its money on mainframes. You heard it first from me: IBM has reached the peak of its development and will decline from now on. Its whole success is based on its name, not its products. But now Big Brother Blue is wearing no clothes. The audience is now looking at the magician's other hand.

The book just published by Richard Thomas DeLamarter (Big Blue: IBM's Use and Abuse of Power, Dodd, Mead, November 10th) will certainly contribute to the disrobing. As senior economist of the US Justice Department in the preparation of the discontinued anti-trust suit against IBM he is an unquestionable authority. He has done an immense amount of homework and gives us a 398-page brief against IBM. His verdict is: Guilty of anti-trust violation by virtue of idiosyncratic and malicious product-design intended to make users IBM-dependent and keep them so.

What they have done to Cromemco is exactly what they have done to all would-be competitors in all fields of computer activity. Theirs was an artificial eminence, and in my opinion they can no longer sustain the image that propped them up.

So we can take comfort in that, at least.

Ben Ross Schneider, Jr.
Lawrence University
Department of English
Appleton, WI 54912

Editor's Reply:

... Your comment that "Cromemco is on the way out unless they admit defeat and use their great creativity and knowhow to make better IBM clones ..." raised my eyebrows. I don't think Cromemco has ever been too concerned about IBM. They don't see themselves as being in competition with IBM for theirs is not the personal computer market.

As early as 1980, as evidenced by the announcement of the multi-user, multitasking Cromix Operating System on the front cover of the premier issue of *I/O NEWS*, Cromemco was moving away from single-user "personal computers." They were concerned with penetrating markets in science & engineering with their state of the art hardware and software development tools.

In short, the "business market" (which became dominated by IBM and spawned legions of clones) has never been a primary target for Cromemco computers. That is not to say that Cromemco computers are not in use running business applications. They are — a great many of them. The development of business application software under the Cromemco development languages and operating systems did occur. Some was absorbed from CP/M, for through some degree of foresight Cromemco made CDOS, and by way of the CDOS Simulator, Cromix, capable of running a good deal of applications software developed under CP/M (dBASE, WordStar, etc.) But it was nothing like the proliferation of software under MS-DOS. As you know, all of that changed when the PC came on the scene. And from what you have stated, the downfall of the Z-80 and CP/M was engineered by Big Blue. But by this time Cromemco was well into their own multi-user, multi-tasking operating system — Cromix. And aside from testing the PC waters with their C-10 (which was too little, too late) they had set their sights away from the PC and towards the minicomputer and mainframe computer with systems that were now being called "supermicrocomputers."

Today, with their state-of-the-art 68000- based line of UNIX systems, it is an exercise in futility to compare a Cromemco System (of whatever flavor) with an IBM PC (of whatever flavor). Cromemco will never develop the often mentioned Cromemco/IBM compatibility bridge (although some enterprising individual or group might). In Cromemco's scheme of things the PC/clone is relegated to being a terminal or standalone workstation connected to a Cromemco multi-user system, and communicating by way of any of a number of available communication packages. Most people believe that UNIX will become the standard operating system, and UNIX software development is expanding at an enormous rate. Cromemco recognized this and offers a very standard port of UNIX System V. They also have an in-house department responsible for 3rd party software acquisition. Part of the work that I do here is directed to that end. After all, it was the vast selection of software that made the PC so appealing. If UNIX does become standard then the question of software availability will become insignificant; the primary factor will be the fitness of the hardware, and that is an area where Cromemco beats IBM hands down. You pointed out that IBM had hurt itself when its own micros began to encroach upon the mainframe domain. If that is the case, then Cromemco has been hurting them all along. In the future they may do it in even a bigger way. And that, too, is something we can take comfort in.

FOR SALE

System III (purchased 1980) with 16FDC, 2 8-inch DD Persci, Tuart, 64K Ram.

3102 Terminal and Keyboard.

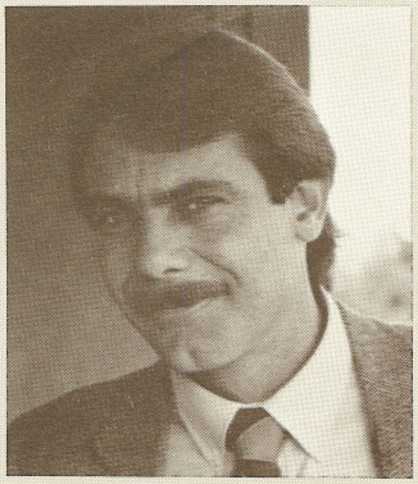
3355A Printer (NEC printer with Cromemco name on it — presently not working).

Plus software, documentation (CDOS, Structured Basic, RATFOR, Wordprocessing Formatter) and manuals. $980.

Quietizer 9821 printer cover ($300 value) for $200.

Contact Valerie Berry
(916) 972-1137

OUTPUT...


Bill Jaenicke

Happy New Year!

By the time you read this the New Year will be upon us. So I will take the time now to wish all of you the best for 1987. What with the flurry of new and exciting developments I'm sure that it will be an exhilarating new year for all of us!

And with 1986 behind us, I want to personally thank the column editors, editorial contributors, the folks at Cromemco, our faithful advertisers, Howard Millman (our art director), my wife Lisa, and all the rest of you who have had a part in making each issue of *I/O NEWS* something special. With similar efforts this year the IACU and *I/O NEWS* are sure to prosper.

Software Resource Guide

One of 1986's projects was the publication of the Software Resource Guide (SRG) for Cromemco Systems. Without a doubt, something of this nature is needed, but many of you voiced a legitimate concern over how updates would be handled, the cost of these updates, their frequency, etc. In short, the format in which the guide was presented obviously was not acceptable.

And so we live and we learn. Rather than abandon the idea, for it is surely something which could be of great benefit to all, we look at other alternatives. 1987 will see something new in the way of a Software Resource Guide. The eventual form it will take will depend largely on your suggestions. What is needed is a format in which additions and changes can be readily made, and to be able to do so in such a way that the costs of updating and distributing the information does not become prohibitive.

In our case, if we were to publish new printed volumes, they would be few and far between because of the high cost of printing and mailing. The same would be true even if the format were such that printed updates could be issued and inserted into a binder containing an existing guide. This would require an updated table of contents and indexes and a peculiar method of numbering pages, all of which means a whole lot of work on someone's part (mine). There has to be a better way. Our computers can provide the way. They afford us the most economical means of distributing this type of information in formats specific to one's individual requirements. For instance, the SRG could be made available in a database format (dBASE, Informix, Unify); it could be provided in an ASCII format (SDF) which could be read into one's own particular database; or it might take the form of an ASCII text file which could be printed or for which SCREEN or CE could be used to "FIND" specified character sequences (types of programs, keywords, etc.). Best of all, it could be set up on the IACU system and accessed via modem.

Lisa Jaenicke

Perhaps all of these options must be employed: what your particular requirements are will depend on your system — C-10, S-100 CDOS, Cromix, Cromix-Plus, or UNIX System V — with modem or without. In any case, something along these lines will be available for IACU members early in 1987.

Where Have all the Gadgets Gone?

Another area that seems to demand some attention, and which I hope will get more coverage, is the realm of gadgetry. Where are the hobbyists and entrepreneurs that create the boards that enable our systems to do some of the menial (and not so menial) tasks that we expect a computer to be able to so. Things like dialing a phone number that we have in one of our databases. Or where's the MIDI interface that let's me plug my electric guitar into my CS-100 and invent and record new sounds? (See *Bits & Bytes*). Its available with other systems — why not ours? After all, Cromemcos are S-100. My guess is that someone has, but hasn't told the rest of us about it. Perhaps they'll talk through the pages of *I/O NEWS* in 1987.

Back to the Present

Though the rest of 1987 holds the unimaginable, this issue has something more concrete to offer. For one, Cromemco introduces its 32-bit wide 2 Megabyte memory board. That means the ultimate in performance with their 32-bit 68020-based XXU. It is showcased with their new CS-120 — an XXU-powered UNIX dynamo.

We are also pleased to present you with an article which both programmers and end-users alike will enjoy: Jeff Johnson's "*How to Write Software for People*." For many years Jeff was in charge of software development for the "Master" series of software at Cromemco, and herein shares some of the philosophy which made programs such as WRITEMASTER and SPELLMASTER so popular.

Jeff also submitted a LISP program solution to the famous "fifteen-puzzle" (that game with the sliding numbered tiles that you would scramble and then try to put back in order). It runs under Cromemco's LISP Interpreter, and graphically displays the repositioning of numbered tiles as it seeks its solution. It is a fine example of game programming using the AI language, LISP.

Jeff mentioned that the program was supposed to have been supplied with Cromemco's LISP package, but never was, and wants to make it available to anyone interested. So, for the time being, if any of you LISP programmers would like the program, just contact *I/O NEWS*.

As the field of Artificial Intelligence is growing in popularity and prominence, I've asked Jeff about putting together an article on the subject. As it stands, the only article on LISP in *I/O NEWS* appeared in the very first issue (1980). If there are any active LISP programmers among you, please speak up. I think there are a lot of Cromemco users interested in what you have to say. And if you like to program, but have never tried LISP, you might want to give it a go. But be forwarned — it can become habit forming.

On a different tack, Dr. Stephen Huber has contributed a fine example of how to conduct a meaningful benchmark with his review of the MAXIMIZER. We hope to provide more articles in this vein in future issues.

In light of the growing interest in UNIX, we are most pleased to introduce a new column which will benefit both Cromix AND UNIX users. It's called SOFT TOOLS and is written by Tom Ronayne, President of Advanced Programming Techniques Corp. This is timely in that many of us find ourselves in transition — though we are accustomed to and enjoy the elegance of Cromix we find it necessary to move on to UNIX. It's not necessarily an easy move to make. For those that are starting out with UNIX it will make a useful complement to Rick Dhaenens' HACKER'S HOME.

The Next Issue

There's a lot in store for you in the upcoming issue of *I/O NEWS*. We'll delve into **CURSES** — the expanded version of termcaps. And we'll have a look at those Persci floppy disk drives through the eyes of John Bush, of Peripheral Labs (former supervisor and lead technician at PerSci Inc.). For those of you with stars in your eyes we'll present the first of a two-part article on "*Microcomputer Systems for Low-Earth-Orbit Satellite Tracking and Data Acquisition*," by Robert J. Diersing, Associate Professor of Computer Science at Corpus Christi State University, Texas.

I'm looking forward to the New Year and to making the IACU and *I/O NEWS* more prominent than ever. Let's do it together.

William E. Jaenicke
Editor & Publisher

System 120

Continued from front cover

the XXU card have already reported phenomenal system performance, and that performance is now available in the System 120. To assess the performance of XXU-based systems in real-life applications, we talked with several IACU members around the world about their experiences.

Dr. Brook Kraeger, a partner in the California-based consulting firm Linsley-Kraeger Associates, was one of the early XXU users. Among other uses for this Cromemco computer, Dr. Kraeger performs water flow analysis through large pipe networks (in order, say, to design drainage systems for large municipalities). Prior to upgrading to the XXU, this analysis typically took 1-1/2 hours to perform on a system with a DPU processor and Maximizer co-processor. With the XXU and its on-board co-processor the analysis took less than 20 minutes. This works out to a remarkable **4.7 times speed improvement** with the XXU.

In Greece, at the University of Padras, Dimitri Pantelatos has reported that two System 400 computers, based on the XPU processor, were upgraded to the XXU. With the new processor these systems showed a **factor of 4** increase in system speed.

In Norway, Svein Rokne had been running finite-element analysis software, which was developed in Austria, on an XPU-based system. By switching to the XXU, Svein reports a **factor of 6** increase in system speed.

From these reports, the new System 120 should show a performance increase of from 4 to 6 times that of the System 100 in many applications. As shown in the table, the System 120 is available in eight different models ranging from $14,995 to $20,995 in price. Cromemco also has available a range of upgrade kits, so that various models of the System 100 can be upgraded to the performance of the System 120.

Continued on page 19

BACK ISSUES
of I/O News are available

C-10 ENCOUNTERS

C-10 ENCOUNTERS is a regular column directed to users of Cromemco's personal computer, the C-10. It is edited by Dr. Tom Beer, of Applied Environmetrics, located at 118 Gordon St., Balwyn, Victoria 3103, Australia. Dr. Beer can be reached by phone during business hours at (03) 817-2571. Submit editorial directly to Dr. Beer.

The correspondence that I get indicates that the level of computer sophistication possessed by C-10 owners and users varies immensely. At the two extremes are the electronics wizard and the software guru. In between lies the typical user who runs application programs, can write a small program in Structured Basic, and is happy enough with that.

Before I got my C-10 I had done considerable Fortran programming, but I had always seen the computer as a mathematical tool to solve problems. I was blithely disinterested in how the computer did its job. All of this changed when the C-10 arrived and the pitter-patter of tiny key-presses filled the house. Having paid a king's ransom for the machine I wanted to make it dance and sing so that I could feel that I had got my money's worth.

I quickly discovered that the price for a dancing and singing machine was an enormous investment in self education. The C-10 manuals provided adequate help for the applications user, but if one wanted more then suitable books and suitable software were required.

Books

Any serious user of the C-10 has to have the *C-10 Technical Reference Manual* (Cromemco Part 023-458). In addition, I purchased Rodnay Zaks book called "How to Program the Z-80" (*Sybex Books*, 1980). This meant that I was almost ready to start programming in Z-80 assembly language. Once an assembler and debugger were obtained then I could get going.

I soon struck a problem. The nature of an assembly language program depends on the operating system under which the program is run. The technical manual gave lots of information about CDOS, (Cromemco Disk Operating System — the C-10 operating system) but it was difficult to understand what it was trying to say. I needed a book about CDOS but of course no such work exists.

Instead I had to make do with works about CP/M and purchased "Mastering CP/M" by A.R. Miller (*Sybex Books*, 1983). This proved to be a good choice and taught me a lot, though I also found it to be hard going. The reason is that I had painstakingly learnt the Z80 opcodes but Miller's book used the 8080 codes. Zak's book has conversion tables in Appendices F and G but it is slow going. Miller's book offers lots of valuable insights on what to do with a macro assembler, though some of the features that he describes are not implemented on the Cromemco macro assembler.

Eventually, I graduated to Johnson-Laird's "The Programmer's CP/M Handbook" which offers lots of examples on the use of System Calls. After getting through all of this material and trying much of it out on the C-10 I have a fair idea of the structure of both CP/M and CDOS and on the implementation of CDOS on the C-10.

CDOS & CP/M

To distinguish the C-10 version of CDOS from the version used on other Cromemco 8 bit machines, the letter C is used in the version number. The C-10 operating systems that have been released since the C-10 was introduced in 1982 have been as follows:

Release 1 CDOS C2.53
Release 2 CDOS C2.56 March 3, 1983
Release 3 CDOS C2.61 July 15, 1983
Release 4 CDOS C2.65 January 4, 1984
Release 5 CDOS C3.07 June 27, 1984

The big change occurred with C3.07 which was no longer supplied on a disk, to be read into memory during the boot startup, but was supplied burnt into a ROM chip inside the innards of the C-10. I am not sure if CDOS Version 3 (which I will refer to as C3 from now on) has been progressively modified, but to the best of my knowledge no new chips have been sent out to SUDS subscribers since the original Release 5 update.

The boot sequence of the C-10 is designed to keep you away from the operating system. Provided MENU.COM and its associated files are on the startup disk then all C-10 versions of CDOS are designed to automatically run the menu. The command CDOS removes one from the menu, and since C2.65, the intrinsic command VER will display the version of the operating system.

CDOS is very like CP/M, but not sufficiently alike that CP/M programs could be guaranteed to run under CDOS. This was a great pity. There was, and still is, an enormous amount of software written for CP/M but only a small amount written for CDOS. Not surprisingly, most C-10 owners wanted more CP/M compatibility. Cromemco met this demand in two ways. On January 31, 1984 they released a disk based CP/M for the C-10. As far as Cromemco were concerned, this was superceded by C3.07 which was billed as being "as fully CP/M compatible as possible." In this column I want to discuss just how compatible is compatible, and point out that there are even occasions when C3.07 is incompatible with C2.65.

Both CP/M and CDOS are disk operating systems. Unfortunately the only standard disk format that emerged was that for 8 inch single-sided

FOR SALE

CROMEMCO SYSTEM 300, 52 meg. HD, 8 inch FD, C-5 terminal & XPU processor, Lots of accessories. Must Sell. $7000 or B/O. (415)592-8055

FOR SALE

Cromemco System Three 4 User System — BRZ-3 Airflow Unit — 2 Cromemco 3102 Terminals — 3355B Fully formed Character printer — Modem & Cable — Writemaster — SpellMaster — dBASE II.

System purchased new for $20K — hardly used. Asking $10,000 o.b.o.

Contact Peter Hall — (704)684-1234
8 — 5:30 EST

FOR SALE

Cromemco System Three (CS3D5E) w/DPU, 512MSU, MCU, PRI, 64FDC, 2-8". Plus C-10 Terminal and Keyboard. Fortran-77, 32K Basic, and more. List $12,570. Asking $3900 or Best Offer.

Call Sarunas Landys(619)259-0319

single-density disks. Each manufacturer went their own way on 5.25 inch formats and this guaranteed that no CP/M disk could be taken, inserted into the standard 10 disk drive and read by CDOS.

Commercial programmers in the CP/M world have tried to get around this by writing disk formatter programs that will allow other machines to format disks in CDOS format and copy files to them. I have not yet seen any that work correctly because the CDOS disk organizes its directory slightly differently from a CP/M disk by putting a disk label as the first directory entry. If you are ever intending to get someone to copy software across different machines then it is wise to supply them with formatted disks (i.e. run COPYDISK on them first), rather than with blank disks and have them try to format them. The difficulties must have been realized by Cromemco as their disk transfer program DiskMaster will not format disks at all.

The screen handling operations built into the C-10 CDOS preclude the operation of certain CP/M programs. In an earlier column I pointed out that the C-10 screen handling is done by including instruction codes at memory address 38H. Any CP/M program that uses those memory locations will blank the screen and will not run on the C-10. There are also more subtle problems that I will now go into.

Applied Environmetrics runs a Family History Research Service and as part of this we market genealogical software. One widely known CP/M package stubbornly refuses to run on the C-10, and I only recently realized the reason for its erratic behavior. During C-10 screen handling operations, the operating system seems to use the program stack to store its return addresses. This means that the stack for a C-10 program has to be larger than the stack for most other machines. The particular program that I was examining has too small a stack and memory locations just under the stack get overwritten. This produces a program that behaves differently each time it is run, because the point during program execution at which the memory location is overwritten depends on the time relationship between the start of program execution and the screen handling cycle.

Both CP/M and CDOS load and run programs the same way. The program is loaded into memory beginning at 100H and execution also begins at that address. For both operating systems the proper way to end a program is to jump to memory location 0, or to issue system call 0 which does exactly the same thing. However it appears that fancy CP/M programmers like to use the fact that the return address to which to jump at the end of the program is held in the HL register of the Z80 when the program starts executing. This is true of C2, but it is not true of C3.

I used to wonder why a professional program like Cromemco's DEBUG had no command to exit the debugger and return to the system. Pressing Control-C works, but leaves files open which can cause problems on Cromix systems using the CDOS simulator. It finally dawned on me that the proper way to exit from the Cromemco debugger is to issue the command to commence execution at memory location 0, which is G 0 and, hey presto, the CDOS prompt appears.

Regular readers of this column and avid scanners of *I/O NEWS* advertisements will know that I am engaged in a process of converting useful CP/M public domain software (PDS) to CDOS. I test all of the programs under C2.65 and if they work then I do not bother to investigate them any further. My attention was recently drawn to the fact that some of the programs on Volume 3 of the PDS series do not work under C3.07. In most cases this was because the program terminated by jumping to the location stored in the HL register on entry — which produces a System Trap under C3.07. Incidentally, after a System Trap message it is vital under C2.65 that the system be re-booted because there is a possibility that the operating system has been corrupted which — at the worst — can wreak irrevocable havoc on your disk. It is also probably a wise precaution to reboot the system after a System Trap under C3.07.

Because I do not use C3 I had to borrow a machine to try the PDS3 programs under C3.07. Making the changes in the last paragraph fixed some of them up, but others still did not work. The reasons were abstruse.

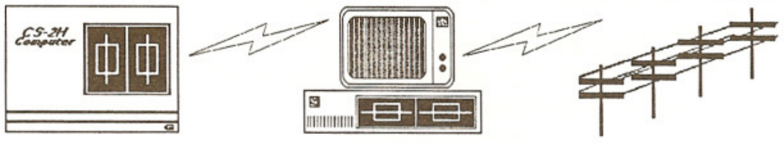
System Call 17 (11h): The technical manual is quite specific about the fact that if 3Fh appears in the first byte of the File Control Block then the current drive is selected. So it is in C2. There is a bug in C3.07 and it will not treat 3Fh as a drive descriptor.

System Call 27 (1Bh): This functions differently to CP/M in all versions of CDOS. Under CP/M this returns the disk cluster allocation map in the HL register. Under CDOS the map is returned in the BC register. Thus any CP/M program using this system call will fail to work under CDOS and has to be fixed up.

System Call 142 (8Eh): This is a system call unique to CDOS because CP/M system calls go no higher than 40 according to Johnson-Laird. Under C2 a cursor address sequence left the A

Cromemco 3102 Terminal Emulator for the IBM PC

With full file transfer capability



ProCall/PC *plus*

from the longest name in Cromemco Software:
ProtoMatrix Software Development

ProCall/PC *plus* for the IBM PC (and compatibles) is the only program that offers you all these features:
• Complete 3102 terminal emulation, including video attributes, editing, and graphics characters.
• DEC VT100 and VT52 emulations also included.
• Supports file transfers with other computers using these major protocols:
Xmodem PcXfer
Kermit Rfile/Sfile
• All assembly language.
• User friendly pop-up menus and dialog boxes.
• Speeds from 110 to 19,200 and 38,400bps.
• Also an excellent general purpose communications program. Use it with BBS systems, telex via EasyLink, news and information services, and more.
• Ability to transfer files with S100 Cromix, Cromix plus, CDOS, C-10 and many other systems.
• Answer mode for remote operation.
• Easy-to-read manual.

And much more, all for under $250.00 (+ tax). ProCall/PC (without emulation) is under $150.00

For more information on ProCall/PC *plus* and other products from PSD, call your nearest Cromemco dealer or PSD at: (408) 263-8665

Or write to us at:
1145 Park Heights Drive
Milpitas, CA 95035
USA.
Telex # 503898

PSD

register unchanged. Under C3 a zero appears in the A register after such a sequence.

Three little, almost insignificant bugs yet each sufficient to change a program that used to run into one that does not. Bugs in operating systems are vexing and inexcusable. CDOS has been prone to them in the high value system calls. Prior to C2.65 the get and set time routines did not work. More annoying was the fact that system calls 160 and 161, the read and write track routines, did not function properly with the write track routine writing a single-density track. The undelete program on PDS3 uses this system call to read in the CDOS directory, undelete a deleted file and write it back to the disk. If used with a CDOS version prior to C2.65 it scrambles the directory and makes the whole disk unreadable. To stop this from happening, the program checks the version and will not continue if it is less than C2.65 — not much help though if you happen to be running your C-10 on one of the old versions of CDOS.

Because I have not implemented C3.07 I have no experience in running 8 inch disk drives with a C-10. If any reader has such experience and would be willing to share their knowledge I, for one, would be thankful. CD

GO WITH THE SPECIALISTS IN CROMEMCO FOR OVER 8 YEARS

2 Megabyte Ram Card with Parity for DPU/XPU/XXU for Cromix & UNIX Systèms	**$1795**
IPC Intelligent I/O Controller Z80 8 Mhz. Processor, 32K two-port Ram I/O Buffer, 0-38K & autobaud supported. Cromix (and soon UNIX) drivers included.	**$1095**

Above boards come with 1 year warranty and 30 day money back guarantee.

OCTART Upgrade from 3.6864 Mhz. to 4.5/5.0/5.5/6.0 Mhz. Operations. Increase throughput by up to 65% (Speedup varies from board to board).	**$ 125**
OCTART Drivers for Cromix (and soon UNIX). Increase throughput by 300% or more.	**$ 195**

Prices subject to change without notice
Dealer Inquiries Welcome

Microcomputer Specialists Inc.
P.O. Box 88127
Grand Rapids, MI 49518
(616)942-5412 or (616)776-9646

You won't find a lower price on Cromemco® — *GUARANTEED.*

Shop. Compare. Then come to the Dildine Industries for all your Cromemco needs.

Give me anyone's advertised price on any current Cromemco® product, and I'll better it or send you a check for the difference when you buy it from me. I'm making this limited-time offer to introduce you to all the services Dildine Industries can provide.

Count on the Dildine Industries

I combine the full spectrum of micro services. One point responsibility for everything from feasibility studies to system integration. And that means less worry and hassle for you. Plus, the kind of courteous service you can depend on.

Call 312-891-1064

If you are tired of trying to find one computer professional who can bring the power of the computer to your business, call me, Kim Dildine, president of Dildine Industries. I believe you'll find my knowledge and expertise a valuable resource. Straight talk, and practical, working solutions to your computer problems.

Dildine Industries, Inc.
P.O. Box 4189
Hammond, IN 46324
312-891-1064

Formerly K.I.D. Enterprises, Inc.
Cromemco dealer since 1978

SOFT TOOLS

SOFT TOOLS is a regularly appearing column dedicated to UNIX and Cromix users. Its aim is towards simplifying the administration and maintenance of multi-user systems. It is edited by Tom Ronayne, President of **Advanced Programming Techniques Corp. (APTC),** P.O. Box 19549, Detroit, MI 48219, (313) 835-0808.

ABSTRACT

The purpose of the new SOFT TOOLS column is introduced by its author, Thomas Ronayne. The main article explores backup issues and strategies: reasons for frequent backup, what's a good backup scheme, desirable characteristics for a backup system, when to do backups, types of backups, Cromix and Unix backup utilities, verifying correctness of backups, and archiving and restoring backups. The backup strategy proposed by the author employs the Cromemco CTD cartridge tape drive and focuses on the uses of the scan.ft214 and ftar *utilities. Command scripts are provided for daily automatic file backup to tape (*daily.cmd *and subordinates).*

An Introduction to SOFT TOOLS

This column — *SOFT TOOLS* — is supposed to do several things — I'm going to try to teach you methods for taking care of your system, making your system easier to use, and making it more efficient. I hope that these methods are practical, efficient, and — wherever possible — elegant. Many of the things I'm going to present aren't "all mine" because I've picked them up from other sources. A good example is this first column — much of the content of the routines presented here is drawn from other sources, thrown in the "blender," and voila!— here's a pretty good way of accomplishing a specific drudgery. This is also a back-handed way of issuing an invitation to you: if you've got some favorite little ditty for doing something, send it along and share it with other folks.

This effort is called "Soft Tools." We were going to call it "Software Tools," but that name was already taken: it's the title of the book written by Brian W. Kernighan and P.J. Plauger (*Addison-Wesely Publishing Company*, Reading, Massachusetts, 1976), that came with Cromemco's RATFOR software package. Ratfor, for those who aren't familiar with it, stands for RATional FORtran, and was my first introduction to the concept of software tools. Software tools are implemented algorithms that are valuable in the production of other programs. Since reading the book — and learning how to use Ratfor — I haven't written a line of "straight Fortran" in years; I only use Ratfor for producing Fortran programs because I've found its tool-like approach to programming far simpler to use and much more "rational" than a straight Fortran approach. We'll explore Ratfor programming (along with C) in many of our efforts down the road. Alas, I take great pride in the fact that I've never written a line of Cobol code in my life (a situation I hope to keep intact forever), so you'll never see any Cobol code. You probably won't see any Pascal code either — I think Pascal is a "pretend" computer language — but you will see some Basic because I think Basic is easy to learn, use, and develop nifty applications with; sort of an "if the shoe fits, use it" attitude.

I'll write about other things too. I'm an opinionated swell old boy (SOB), and I tend to call spades spades when I think they are. I'll tell you what I think about software, hardware, books I read, articles I read, and any other thing that may strike my fancy. I'll also tell you why I think a certain way about things. I don't claim infallibility; rather, I'm reacting against a tendency I see in the literature to "Marvin Milktoast" any- and everything.

I also presume that you're able to read manuals — we won't get real basic in these things — but I'll point you in the right direction if something appears to be esoteric and needs further explanation.

If you ever flew a kite as a kid, you'll remember the hopeless knots you'd get every so often in your string ball. Well, bringing up UNIX is a lot like that, and we're going to explore the intricacies of UNIX in future columns — try to shed some light on things, as it were. Again, much of this exploration will be based on my own and other's experiences.

So, who am I? Well, I'm founder, chief programmer, bottle washer, toilet bowl cleaner, president, and whatever of **Advance Programming Techniques Corporation (APTC)**, Post Office Box 19549, Detroit, Michigan 48219, (313) 835-0808. APTC is a Cromemco dealer here in Detroit. We sell and service Cromemco computers, we provide custom programming, and we try to stay out of trouble. We concentrate on the so-called "heavy-duty" areas; i.e., scientific and technical programming (need a good Gaussian routine — give me a call), data base design and implementation (we consider ourselves Informix experts), and we're actively marketing ColorGraphics Systems' ArtStar computer graphics system to corporate art departments and graphics houses in this area.

I go back a way's with computers — I started in 1961 with IBM punch-card equipment programming (when programs were "written" with jumper wires plugged into jacks on large boards; stuff like the 402, 407, things like that). I graduated to the 1401's, then to a Honeywell 600 (it had real "core" in it), then to Honeywell 66/6000's in the "mainframe" world. I bought one of the first Altair kits, built it, sold it to a guy (who's still using it and won't sell it back to me), and went through Z80 assembly and other foolishness on the way to really beginning to understand Cromix and UNIX.

I've watched hardware change from — in the microcomputer world — programming by flipping switches to today's truly incredible multi-user, multi-tasking operating systems: you can, without a lot of trouble, support 128 users on a Cromemco System 420 — a MICROcomputer — for crying out loud!

I hope I've learned a few things along the way, and, like I said earlier, I'll try to pass some of them along. To borrow a phrase from the mayor of New York, let me know how I'm doin'.

Backup Issues and Strategies

How much would it cost you if you lost a file, a file system, or a disk? How much time would it take you to recreate them? What is your time worth? How you answer determines what backup strategy you should use for your computer system.

In a high-performance, multi-user Cromix-Plus or Unix System V computer system (and on single-user systems, too!), users have been known ("No, really?" 'Trust me.") to accidentally erase files.

System administrators can accidentally destroy files (or a complete file system). A disk drive can become unusable through mechanical or electrical failure. A lightning strike can cause havoc in the best-isolated system (the effects of lightning can be minimized, but there is no known way to eliminate all damage when millions of electron volts start dancing around in your wiring). Backups can help minimize losses in cases like these.

A backup scheme will help you recover either a file or a complete file system when something happens — bear in mind that the key word here is "when," not "if:" You can count on something or somebody blowing it, and you'd be miles ahead if you have your "iron underwear" in place before it happens.

This article discusses backup schemes that have been found, in practice, to be both effective and mostly painless: We'll discuss characteristics of good backup schemes and the Cromix and Unix utilities that make and restore backups.

What's a Good Backup Scheme?

A good backup scheme has some typical properties:

- It's tolerant of media failure
- It's easy to use
- It can provide a list of what's on a particular tape
- It's portable
- It verifies that a backup worked
- It assures that backups are stored remotely from the computer system so a disaster won't take both the computer system and the backup

Note that we said "tape." Backup can be done on many different media: floppy disks, fixed disks, magnetic tape (in many different forms — a floppy disk is, after all, magnetic tape in a slightly different form), punched cards, punched paper tape (remember these?), and, recently, optical disk and Cauzin Systems' Softstrip™ System (a printed, machine-readable paper strip).

We don't really recommend anything for Cromix and Unix systems except magnetic tape — either 9-track reels or tape cartridges — it is, right now, the cheapest media (cost-per-byte of storage, equipment, maintenance, and operator time). A formatted tape cartridge holds about 25.7M bytes of information, costs about $30, and can be stored in a drawer. 9-Track tapes cost about the same as a tape cartridge though it holds more, but the equipment is about three times more expensive, operator time is needed, and storage of the tape reels requires expensive floor and shelf space.

The other backup media mentioned share several undesirable features: they're slow, they require much operator time, they can't be automated, the equipment is expensive, the media are expensive, and so on. The two biggest items are operator time and automation; e.g., backing up a single 20M byte hard disk partition would take about 52 floppy disks (assuming the partition is full of files), and an operator would have to sit at the machine to change disks every three or four minutes (that translates to about three hours).

This leads us to a check-list of desirable characteristics for a backup system:

- It minimizes the time spent doing the backup
- It minimizes the time the operator has to spend doing backup
- It minimizes the machine time for backup (backup utilities degrade system performance by grabbing almost total control so files aren't opened during the backup time)
- It minimizes the time needed to restore a file or group of files (imagine (or remember) searching through 30 or 40 floppy disks looking for one file)
- It minimizes both the size and redundancy of backup by backing up only files that have changed
- It minimizes the amount of disk storage area necessary to do a backup
- It provides frequent backup so that the backup is always as current as it can be

All these desirable properties and characteristics can't be optimized at the same time; some of them are mutually exclusive. It's up to you, at your site, to figure out what particular thing(s) should be given the greatest priority, next greatest, and so on based on the resources available and the requirements of your user community. Here are some things to think about, based on the most-asked questions we get regarding backup schemes and strategies:

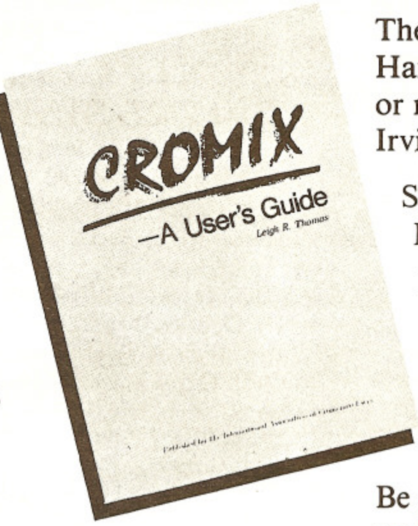
Question: *What if my tape goes bad?* Tapes do go bad: they wear out, they develop spots that can't be written to or read from, they get zapped by stray magnetic fields (don't put them near motors, for example). It's a good idea to have two copies of backup files. Making multiple copies isn't that difficult (see below), and the odds of having both of them go bad are low.

Attention CROMIX Users

The book you've been expecting is finally here.

CROMIX — A User's Guide can be ordered now.

(While They Last)



Leigh Thomas' excellent tutorial on CROMIX, reprinted in the U.S., can now be ordered directly from I/O News.

The price is $25.00 plus $4.95 Shipping & Handling per copy. For bulk orders of 10 or more, the price is $17.75 per copy F.O.B. Irvine, California. California residents add 6% sales tax

Send Check, Money Order, or Credit Card Information to:

I/O News
ATTN: CROMIX — A User's Guide
P.O. Box 17658
Irvine, CA 92713
U.S.A.

Be sure to include Membership Number and full shipping instructions with order. (Amounts noted are in U.S. dollars drawn on U.S. banks, only.)

This is a MUST BOOK for any CROMIX user.

Question: *When should I throw a tape away?* Tapes and floppy disks do wear out. Eventually they have to be replaced, and they should be replaced before they wear out and "gotcha." It's a good idea for your site to keep archives, and retire the oldest tapes into the archives. This serves two purposes: you retire your oldest tapes into the permanent archive ("permanent" can be from three to seven years, depending on your needs and any possible legal requirements), and fresh tapes become part of the "backup pool" that eventually migrate to an "oldest tape" status when they're retired. Here's an important point: archives are kept forever — see "permanent' — backups are constantly over-written with new backup data.

Question: *Can it be easy to use?* Sure: making backup tapes and restoring files can and should be so simple that the task won't be avoided (as it frequently is when floppy disks are used for backup) because of complexity or time required. The "trick" is to automate — as much as possible — the task, and to make the computer do as much of the work as possible. The "daily" script below is one such way.

Question: *Is it easy to recover a single file?* Backup tapes are used to recover files far more often than to recover file systems. It should be easy for an operator to determine what tape a particular file is on, and it should be "no great trick" to recover that file.

Question: *Is it self-checking?* You should never wait for a disk to go bad to find out that your backups aren't readable. The backup scheme should be self-checking, automatically.

Question: *Is it portable?* Backups are seldom used, and, when they are, they're usually used on the same machine that created them. If disaster strikes or you upgrade your machine, you'd better be able to read your backup media. An additional wrinkle comes into play with Cromix and Unix — many sites use both operating systems, and it's nice to know that you can transfer information (not "runnable" files, but any text files) between machines via the backup media.

When To Do Backups

You should make backups when the system isn't in use, or when any activity on the system is minimum. This is because files that are in-use may change when you're doing the backup, and you'll have a "bastard" copy. We strongly recommend that backup take place in the middle of the night. We go even further — the "daily" backup script warns any users on the system (at midnight) that it's going to do a backup, gives them a chance to log off the system, and then arbitrarily hangs-up on anybody bold and brassy enough to stay on.

This is not arbitrary — it's iron underwear: if a user is on the system, you can assume (even if somebody just forgot to log off) that a file might be changed during the backup procedure. You don't want that to happen, so you prevent it by logging everything off, putting the system in single user mode (so nobody can log on), and proceed with the backup work.

An alternative is if you have excessive disk space available. You can make a backup on an un-used disk partition (which is fast because disk-to-disk file moves occur at or near the streaming rate of the drive and controller), and then copy it to tape later. You take the chance, though, that you won't have a disk drive problem, so we tend to discourage this in practice.

To belabor the point: how much can you afford to lose? Backing up to an unused partition (or an unused disk drive) is fast and convenient, but, what if something happens?

Types of Backups

Backups fall into two categories: full and incremental. Full backups are complete; you back up a complete file system, but they take the most time to do, and typically should be done once a month for each complete file system (a complete disk partition).

Incremental backups, on the other hand, takes less time and tape. In an incremental backup, you only back up files that have changed since some arbitrary date. As it happens, Cromix and

CROMEMCO'S LARGEST DEALER IN THE U.S.A.

THERE MUST BE A REASON!!!

CALL (214) 231-6108
P.O. Box 832117
Richardson, TX 75083-2117

SOFTWARE
SERVICE
HARDWARE

COMPUTER CROSSROADS of AMERICA, INC.

Where It All Comes Together

Unix keep track of dates in a way useful for incremental backups. The operating system keeps track of a files' created, modified, accessed, and dumped dates. Every time you access a file, the accessed time and date is changed ("running" a program does not "access" it — changing it does). Every time you make a backup, you can "touch" the dumped time and date with the backup time and date. Then, it's simple to just compare the dumped and accessed dates and back up only files that have a accessed date greater than the dumped date.

In this way, you keep incrementing the size of the backup tape (and the number of files) until you reach another full backup time — then every file is backed up, the date and times change, and you're ready for another accessed/dumped cycle.

Cromix and Unix Backup Utilities

Cromemco includes multiple utilities for making backups: the rcopy utility will make a bit-for-bit copy of a device (floppy or hard disk) on another device; tar which provides multiple ways and means for creating and retrieving backups and archives on multiple media; ftar which works much like tar.

We're going to concentrate on the ftar utility — it does the things we want to do, and it reads and writes from floppy, tape cartridges, and disk partitions. It may seem that we're dismissing rcopy and tar out-of-hand; we're not — they're harder to use, and we want to keep this both useable and simple. We're also going to concentrate on the model **CTD Cartridge Tape Drive** — it's cheap, it's easy to use, and — so far — we've found it reliable for backup and archiving.

Rcopy is useful for copying "whole things." You can use Rcopy to make a bit-for-bit copy of something onto something else (you can, for example, make an exact copy of a MicroSoft MSDOS disk onto your Cromix disk and then make an exact copy of it to a blank).

Tar is mainly intended for 9-track tape, and has an advantage in that the tapes it creates can be read and written by any other Cromix or Unix system using tar. The current release of ftar (30.12) can write tar-format media, and it's easier to use, which is why we're going to focus on it.

We also will concentrate on the scan utility (which, in Cromix, has replaced the find utility), which we will use as a pipe and filter mechanism to feed file names into the ftar utility.

Here is how to use the scan utility to find files that have been modified since the last time they were dumped to an archive tape:

```
scan / 'type = = is_ordin && tmodify >   tdumped && print(path)'
```

which will print the complete path name of a file.

Here is how to use the ftar utility to take a list of path names from stdin (your console) and write them to a CTD tape cartridge: ftar -ic -b 510 /dev/ftcd

Note: the "-b 510" defines the size of the file buffer ftar will use when reading and writing files from the device — if you have a megabyte or less of RAM, you'll get an "out of memory" message with this option when there are things running in background. If you choose to use the suggestions here, and have a problem, just eliminate the buffer size option.

Here is the most important thing — this command line (which is one line in the system, but printed on two lines because of the length of it) that scans the system for files modified since the last dump date and pipes the name into the ftar utility for writing to the tape (quick, simple, and, thank you, elegant):

```
  scan / 'type = = is_ordin && tmodify> tdumped && print(path)'
| ftar -ci -b 510 /dev/ftcd
```

This is the heart and soul of the incremental backup script, "daily" presented below (we didn't make this up, Cromemco supplies it in the help file on scan — you've got to dig, but it's there).

Doing a full backup is easier:

```
ftar -cd -b 510 /dev/ftcd {pathname}
```

That command line is a bit of a misnomer: if you make a full backup of, say, the root, you're tempted to use:

```
ftar -cd -b 510 /dev/ftcd /
```

which is just fine if you're going to restore to the root (ftar will blithely create "/" and all its sub-directories for you). If, on the other hand, you might want to restore to "someplace else," you should enter this:

```
ftar -cd -b 510 /dev/ftcd **
```

from the root. The "**" won't give you a "too many arguments" error because your root doesn't have too many file names, and you can then use ftar to recover the files on another device without worrying about writing into the root; i.e., ftar will recover what's on tape to "here" — the current directory. You need to give some thought to "where" you might want to restore things, and you really should use absolute path names in most instances.

There's no need to scan for anything, and the "d" in the command options updates file dumped dates.

This procedure assumes that your entire file system won't be modified on a daily basis — if you see more than 25.7M bytes' worth of files being modified between full backups, you'll want to use a different procedure which performs incremental backups on selected partitions (it's not presented here).

You should use the "usage" command — with every partition mounted — to determine exactly how much tape you're going to need for either an incremental or archive backup. If usage tells you that you've got more than 25.7M, you need a different approach, or you've got to do it by hand.

We think that it's valuable to have a printed copy of what got backed up, so we add an option and re- direction to these commands: by including the "v" (for verbose) option and by re-directing the verbose output to, say, the system printer, you have a printed copy of what got backed up when:

```
ftar -cdv -b 510 /dev/ftcd {pathname} > /dev/prt
```

QUINN TEAM

I N C O R P O R A T E D

CONGRATULATIONS !

QUINN TEAM HAS BEEN BOUGHT OUT!!

DATA AIDS of Houston, Texas has purchased Quinn Team's service department. DATA AIDS specializes in data communications and networks.

Rich Quinn will be moving in January to continue his pursuit of excellent service FOR DATA AIDS.

YOU CAN STILL receive expert **CROMEMCO** service by **RICH QUINN** through **GUNN ENTERPRISES** in Houston, Texas.

You will **STILL** have expert advice in **TEC TIPS** from **RICH QUINN**.

A new era is beginning for QUINN TEAM

Quinn Team. Here to serve you with the best products yet.

(818) 889-4819 *30313 CANWOOD STREET AGOURA • CALIFORNIA 91301*

By the way, you do not want to pipe to the spooler — you'll have an "un-allocated i-node" error when ftar is doing its thing and trying to write the file names to a file at the same time; the system loses its alleged mind.

Inittape

Before you can do anything with any magnetic media, you've got to initialize it. Cromemco provides the inittape utility (in both Cromix and UNIX), but it can be a pain to use.

You can use inittape the same way you use initflop; just type the command and go get a cup of coffee. If you've got time, this works, but a better way is like this:

```
inittape -df -p 1 /dev/ftcd
```

This says, "initialize the tape in detached mode, don't wait for retension, do one verify pass, and do it on /dev/ftcd."

Why? Stick a tape in the drive and wait for it to do a re-wind before you enter the command — tapes have been know to come off their hubs. Detached mode is so you don't have to sit there and wait for the "edit VTOC" question (inittape automatically writes the volume table of contents for you). One pass should be enough — if you get too many errors, send the tape back to 3M for a warranty replacement.

Inittape pretty much takes total control of the machine — nobody else can do much when it's running, so you might want to do it at lunch time (the command given here takes about twenty minutes to run). You can, if you'd like, do this as a first procedure in the automatic backup given below, but we don't recommend it — you do need to look at things every so often.

Partitioning

This is probably as good a place as any to discuss partitioning. Those of you who have Cromix-only machines may have partitioned your Winchester (or other) hard drives in some way that may or may not be conducive to making backups. Those with UNIX/Cromix machines received the drives with Cromemco's standard partitions, which are the way they are for a real good reason: the factory-standard partitions are the size that fits on a CTD cartridge.

We've seen machines with 50M byte Winchester's with only one partition; i.e., the whole 50M available. We tend to discourage that.

We've come to believe that the "best" partitioning scheme is the one used in the UNIX/Cromix machines; std0 is about 3000K, std1 is about 4200K, std2 is about 20100K, std3 is about 20100K, and std4 is somewhere between 1350K and 1800K, depending on what hard drive you have.

In a Cromix-only environment, we like to combine std0 and std1 (making a new std0 that's about 7200K), make std1 20100K, std2 20100K, and make std3 whatever's left over. We put all the operating software in std0, except for the /usr directory — we put /usr in std1 and mount it on boot to /usr (we do the same thing in UNIX systems, except that std3 is mounted to /usr).

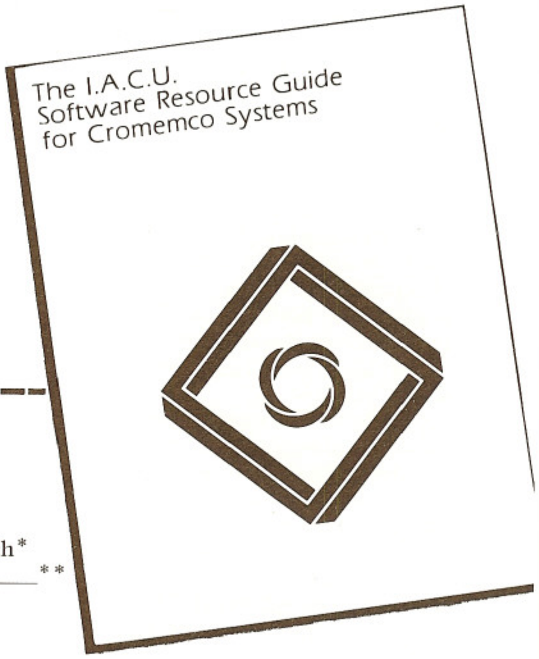
We do this because there's no real good reason to constantly archive std0: it contains Cromix, /bin, /cmd, /etc, /gen and other directories that you only change when you upgrade the software (you do subscribe to SUDS, don't you?) You make one archive tape, put it away, and only use it when necessary. Note that the script given here violates this rule, but it's intended

---

Having problems locating software?

Now there's help ...

The IACU Software Resource Guide for Cromemco Systems


The I.A.C.U.
Software Resource Guide
for Cromemco Systems

☆ Over 180 pages

☆ Multiple indices - by product name, application group, operating system, etc.

☆ Complete product descriptions — from source language to technical support

Order your copy now!
Only $50

(Shipping Charges: Over $2.50 U.S. will be invoiced. International orders must specify Air or Surface Mail and will be invoiced.)

(California residents add 6% sales tax)

The I.A.C.U.
P.O. Box 17658
Irvine, CA 92713
U.S.A.

Send __ copies @ $50.00 each*
I enclose $__________ **
☐ Check ☐ Money Order
(make payable to The IACU)

Charge my ☐ MasterCard ☐ Visa
Phone orders with credit card call (714) 661-9764

Card No.__________ Expires__________
Signature__________

(Please Print)
Name __________
Address __________
____
City __________ State ______ Zip ______ Phone __________
Country __________

* Calif. residents add 6% sales tax
** U.S. Dollars. International checks not drawn on U.S. bank add $10

as a guide for you to do your own thing, so it does back up from the root; if you partition like we do, just change the scan and ftar lines to point to the right place — instead of "scan / ...." use "scan /usr ..."

We take pains to move the compilers and things out of /usr/pkg/whatever to the /bin, /cmd, and any other directories and to move any "maklink" files to std1 or std2 (because you can't make a link across devices).

Then, you can use std2 or std3 for an "on-line' back up, and ftar whatever is in them to tape — with the proviso that this may not be a good idea, see above.

Restoring

Restoring a file from tape is simple:

```
ftar -xv -b 510 /dev/ftcd {pathname}
```

gets a file (or group of files) back and shows you the names as they're recovered.

Restoring a whole file system is different:

```
ftar -xv -b 510 /dev/ftcd
```

Did the Backup Work?

The procedures given above write files to a tape cartridge. The ever-present question is, did they write properly? The "proof of the pudding":

```
ftar -yv -b 510 /dev/ftcd
```

which compares what's on tape with what's on the file system, and prints (on the console) the results of the compare (or, by re-direction, to the system printer, see above, or to a file). You should make the comparison on both an incremental and full backup tape. Period. If you don't, you're asking for it.

If your media is bad — the tape cartridge — you'll get a "UFLOP error." A UFLOP error means that ftar couldn't read a spot on the tape. This means that you should re-initialize that cartridge, which will create a new VTOC for you. It also means that that particular cartridge should probably be retired to long-term archive or scrapped.

Which Tapes Do You Restore?

The simplest strategy we use or recommend is a full backup that occurs after Friday's business; i.e., about 00:00 Saturday (this is how the daily script is written). Then, about 00:00 Tuesday, Wednesday, Thursday, and Friday, we make incremental backups. In a simple (simple-minded?) implementation of this particular scheme, you make an archive tape on Saturday, replace the cartridge on Monday morning, and do incremental's through the week on the replacement cartridge. Then, on Saturday, a new full archive is written and the whole cycle starts over.

This is NOT a good way to do things — a different backup cartridge should go into the drive every business day, and the archive tape should be archived. It is, however, cheap — you only need two tape cartridges. It is also taking a chance: 3M DC-600A cartridges have a sad history of either being bad out of the box or going bad shortly after being placed in service. If you don't use the inittape utility every so often (like once a week) to check a cartridge in a two-tape system, you're asking for trouble; i.e., you may find your backup unreadable. Lesson: tapes are cheap; buy some.

So, how do you restore? Depends. If you're only restoring a particular file that somebody blew away, you can see if it's on "yesterday's' incremental tape, then on the day's before, and so on back to the last full save. This is done with:

```
ftar -tv -b 510 /dev/ftcd {pathname}
```

which will tell you if it's there, and:

```
ftar -xv -b 510 /dev/ftcd {pathname}
```

to get it back.

If, on the other hand, you have to restore a full partition, you start with the last full save and work toward "yesterday" tape-by-tape. This guarantees that all files that were in a directory are brought back (from the full save), and that any incremental changes made will over-write in the proper sequence. How many tapes you have to read depends on when you did the last full backup (we recommend a weekly full backup).

This is really the difference between tar and ftar. Tar can append files to an archive: ftar can't. I'd prefer an ftar tape to work with because I can easily re-load without having to worry about possible appends down the road. Ftar is a *wysiwyg* (what you see is what you got); you don't have to fool with it.

An additional advantage of ftar (we believe), is that ftar will re-create devices and directories for you — tar won't — so a full partition restore is much, much easier because you don't have to re- create all the directory structure before you can get back in business.

So, What's a Good Strategy?

We believe that the "best" strategy is one in which you create daily backup tapes that "grow" as the week goes by, an end-of-week full file system save, and an end-of-month file system archive.

The way we believe is best to implement this strategy is this: everything works on a "rotating FIFO" basis (FIFO is an accounting acronym that comes from first-in-first-out).

You should have four daily tapes, labeled "Monday," 'Tuesday," 'Wednesday," and "Thursday." On Monday, you put the "Monday" tape in the drive, on Tuesday you put the "Tuesday" tape in, and so on. You rotate these tapes every week, using them for about three months before you really start paying attention to using inittape to verify them.

You should have four weekly save tapes, labeled "One," 'Two," etc. or some other labeling that makes sense to you. You use the number one tape on the first Friday of the month, the number two tape on the second Friday, and so on through the last week of the month.

At the end of the month, you should have a tape (or tapes) marked with the month and year, ftar everything to them, and put them away. We recommend that you "put them away" somewhere off the premises (your safe deposit box is a good place) so if a real catastrophe strikes — like the building burns — you have a known good copy of your system that can get you back in business as soon as possible. (Does this sound like over-kill? It isn't: it's happened; consider, again, what it would cost you if you lost everything to a fire, flood, or earthquake.)

Automated Daily Backup

Here's a routine we've cobbled together over the years that incorporates suggestions from others, things printed in *I/O NEWS*, the "help" files in Cromix, UNIX manuals, special interest magazines, and our own experience. It's called "daily" (daily.cmd, **Listing I**), and we have it living in the /cmd directory and started up on bootstrap running in background like flush; e.g.,

```
daily &
```

in the /etc/startup.cmd file.

There's nothing esoteric about this: it freely uses Cromix shell capabilities.

As you study daily, you'll notice that there are "echoes" to /dev/console, and you'll probably think to yourself that these are unnecessary because any messages will be directed to the console anyway because it's the standard output and standard error device. True; but, we recommend that you change the "console" to "prt" so you have a printed record of what happened, when. (We also suggest that you edit form-feeds into the strings so you have a neatly paged record to file with your archive tapes.)

One word of warning — if your file system is bigger than 25M bytes, ftar will prompt you to change the tape when it runs out of space, and the whole system will sit and wait for you to do so. You may wish to modify this routine to handle that (we have, a multi-partition version is available, see below).

Listing II is stopflush.cmd, which should go in your /cmd directory. It's used to kill the flush utility after the appropriate process ID (PID) is appended to it.

The commands that hang-up any users still on the system when backup starts aren't included in **Listing I**. This requires a program — **Listings III** — and may be beyond the needs of most sites. If you're interested, we'll be happy to make a copy of the source and .bin files (including the files listed here) for you if you send us a blank disk (5-1/4'' or 8'') and $5.00.

We also maintain a ''guest'' account on our system — you may, if you'd like, dial-in to us and copy these files. You may dial in at 300, 1200, or 2400 baud to (313) 835-0809, login as ''guest,'' and ''rfile/sfile'' all the files (if you have trouble, send mail to ''system'').

Conclusions

Making backups of your system is one of the most important activities, and most often neglected, you need to do. We believe that ''daily'' is a painless, inexpensive, and easy way of handling this task, and we urge you to take advantage of it on your system.

LISTING I

```
\       daily--perform daily and weekly archive to floppy tape
\
\       initialize /etc/.day2 file for this shell routine startup
day > /etc/.day2
\
\*******************************************************************************
\
\       wait--check for change in the day of the week
\
\wait
\
\       put the day in file /etc/.day1 for comparison
day > /etc/.day1
\       compare /etc/.day1 and /etc/.day2 for match, result in /etc/.day3
cmpasc /etc/.day1 /etc/.day2 >* /etc/.day3
\       get the first line of /etc/.day3
input < /etc/.day3 > /etc/.day1
\       if the first character is "/", there's no change of day
testinp -f /etc/.day1 /
\       the day did change, transfer to the backup routine
if -err goto backup
\       update /etc/.day2 with the current day-of-the-week
day > /etc/.day2
\       put the process to sleep for 90 minutes
sleep 5400
\       check for roll-over to new day-of-the-week
goto wait
\
\*******************************************************************************
\
\       backup--check day of week, transfer to appropriate archive commands
\
\backup
\
\       first, clean out .bak, .prn, and file corpses
echo "Cleaning out the file system" > /dev/console
echo > /dev/console
time > /dev/console
scan / '(ext == ".bak" || ext == ".prn" || ext == ".sbk" || ext == ".i" || ext
== ".out") && shell ("del -v "|path)'
echo > /dev/console
free > /dev/console
echo "File System Cleaning completed" > /dev/console
\       update /etc/.day2 with the current day-of-the-week
day > /etc/.day2
\       test for what day it is, take appropriate action
testinp -r /etc/.day2 "Today is Monday."
if -err goto none
testinp -r /etc/.day2 "Today is Tuesday."
if -err goto regular
testinp -r /etc/.day2 "Today is Wednesday."
if -err goto regular
testinp -r /etc/.day2 "Today is Thursday."
if -err goto regular
testinp -r /etc/.day2 "Today is Friday."
if -err goto regular
testinp -r /etc/.day2 "Today is Saturday."
if -err goto biggie
testinp -r /etc/.day2 "Today is Sunday."
if -err goto none
\

\*******************************************************************************
\
\       biggie--make a file system archive tape
\
\biggie
\
echo "Performing Total System Backup" > /dev/console
time > /dev/console
\       unmount ramdsk
unmount rd0 >* /dev/null
\       kill the flush utility
\       get system status, match for "flush", re-direct into temporary file
ps -ab | match flush >* /tmp/temp
\       put the command line in a file
echo -n "stopflush " >* /tmp/temp.cmd
\       concatenate the line from first file to the second file
type /tmp/temp >> /tmp/temp.cmd
\       execute the kill
/tmp/temp
\       clear out the temporary files
del /tmp/temp /tmp/temp.cmd
\       retension the floppy tape cartridge
mode /dev/ftcd reten -1
\       set floppy tape parameters
mode /dev/ftcd verify reten 0 retry 10
\       flush the system buffers, just in case
sync
\       ftar the root to the floppy tape
ftar -cdv /dev/ftcd / > /dev/console
\       verify the backup
ftar -yv /dev/ftcd > /dev/console
\       re-mount ramdsk
mount rd0 /ram >* /dev/null
\       re-start the flush utility
flush 600 &
\       finished, jump back to control loop
goto wait
\
\*******************************************************************************
\
\       regular--perform incremental backup of files that have been changed
\
\regular
\
echo "Performing Regular System Backup" > /dev/console
time > /dev/console
\       unmount ramdsk
unmount rd0 >* /dev/null
\       kill the flush utility
\       get system status, match for "flush", re direct into temporary file
ps -ab | match flush >* /tmp/temp
\       put the command line in a file
echo -n "stopflush " >* /tmp/temp.cmd
\       concatenate the line from first file to the second file
type /tmp/temp >> /tmp/temp.cmd
\       execute the kill
/tmp/temp
\       clear out the temporary files
del /tmp/temp /tmp/temp.cmd
\       retension the floppy tape cartridge
mode /dev/ftcd reten -1
\       set floppy tape parameters
mode /dev/ftcd verify reten 0 retry 10
\       flush the system buffers, just in case
sync
\       ftar only files that have been modified to floppy tape
scan / 'type == is_ordin && tmodify > tdumped && print(path)' | ftar -civ
/dev/ftcd > /dev/console
\       verify the backup
ftar -yv /dev/ftcd > /dev/console
\       re-mount ramdsk
mount rd0 /ram >* /dev/null
\       restart the flush utility
flush 600 &
\       finished--go to control loop
goto wait
\
\*******************************************************************************
\
\       none--no backup performed on selected days
\
\none
\
echo "No System Backup Today" > /dev/console
time > /dev/console
goto wait
```

LISTING II

```
\       stopit--kill the flush utility
\
\       get system status, match for "flush", re direct into temporary file
ps -ab | match flush >* /tmp/temp
\       put the command line in a file
echo -n "stopflush " >* /tmp/temp.cmd
\       concatenate the line from first file to the second file
type /tmp/temp >> /tmp/temp.cmd
\       execute the kill
/tmp/temp
\       clear out the temporary files
del /tmp/temp /tmp/temp.cmd

\       stopflush--kill the flush utility pid from stopit.cmd
kill -3 #1
```

LISTING III

```
\       logout--log any active users off the system
\
\       put out the message to everyone logged-in
msg -a < /etc/.logout1
sleep 60
msg -a < /etc/.logout2
\       get the system status into /tmp/status
ps -al >* /tmp/status
\       find out who--if anyone--is logged on the system
who >* /tmp/who
\       run the hangup program
echo -n 'Time to forced logout: 5 '; sleep 1
echo -n '4 '; sleep 1
echo -n '3 '; sleep 1
echo -n '2 '; sleep 1
echo -n '1 '; sleep 1
echo -n '0 '; /bin/hangup
echo
\       clear out the temporary files
\ del /tmp/status /tmp/who
```

PERSCI 299-B

Reconditioned	$295
30-day warranty	
As-is	$95

No missing parts

ADX

P.O. Box 24168
Lyndhurst, OH 44124
(216) 486-2888

BITS & BYTES is the place to look for the odd bit of information, opinion, programs, profiles and rumors that circulate through The IACU. Our ears are always attuned to any interesting miscellany — if you have something to contribute send it along to I/O NEWS • P.O. Box 17658 • Irvine, CA 92713 • (714) 661-9764

Cromemco Merges With Dynatech

Dr. Harry Garland, president of Cromemco, has announced the merger of Cromemco, Inc. with Dynatech corporation. Dynatech is a publicly-traded Fortune 1000 company with headquarters in Burlington, Massachusetts.

In announcing the merger Dr. Garland said, "This new association with Dynatech will bring new visibility and stability to Cromemco and will augment our considerable strength as a technology leader in the design of specialized, high-performance supermicrocomputer systems. While many of the early microcomputer manufacturers have foundered during the past few years, Cromemco has continued to develop new markets and set new price/performance points for the industry. Now, as part of a large, stable and diversified company, our customers can look for Cromemco to continue to serve their specialized computing needs for many years to come."

Dynatech was founded in 1959 by Warren Rohsenow and J.P. Barger, both of whom were on the engineering faculty of MIT. In the mid-60's Dynatech began a strategy of diversification through the acquisition of high-quality, specialized technology companies. Today there are over 30 companies in the Dynatech organization engaged in medical electronics, digital communications, diagnostic instruments, and other high-technology niche markets. Both founders continue to manage the company with Warren Rohsenow serving as chairman, and J.P. Barger as president. The success of Dynatech's strategy is evidenced by Dynatech's recent listing as one of the 101 best performing companies in America.

With the new association with Dynatech, Cromemco is well positioned to continue its decade-long commitment to its customers, while continuing to develop new and innovative products for the future. Cromemco will continue to operate from its Mountain View headquarters and expects no significant internal changes as a result of the merger.

XXU Technical Bulletins

Cromemco has released three technical bulletins which may be of interest to those contemplating upgrading their DPU/XPU system to the new XXU. The technical bulletins describe the modifications to convert an STDC-XPU/DPU (for use with XPU or DPU systems) to an STDX-XXU/XPU (for use with XXU, XPU, or DPU systems), an MCU (for use with XPU or DPU systems) to an MCUX (for use with XXU, XPU or DPU systems), and a 64FDC-X (for use with XPU systems) to a 64FDX (for use with XXU or XPU systems)

In addition, a technical bulletin is available to correct the incorrect Octart interrupt vector numbers in the /usr/sys/cf/dfile for even-numbered Octarts running under UNIX System V.2 release 1.02. Until corrected, even-numbered Octarts 2, 4, 6, and 8 will not function.

Copies of these technical bulletins can be obtained from *I/O NEWS*.

MIDI for Cromemcos?

IACU member Jack E. Burkett, of Houston, Texas, recently wrote an inquiry regarding the availability of a MIDI interface for Cromemco computers (a C-10 in particular). As I did not know what a MIDI interface was, I wrote back for an explanation. The reply was, "MIDI means Musical Instrument Digital Interface. It is the new 31250 baud Asynchronous TTL standard for microcomputer communications with digital pianos, synthesizers, etc. The International MIDI Association has set the standard for 10-bit codes which enable computers to play musical instruments. See BYTE, June 1986, Vol. II No. 6.. Also KEYBOARD magazines or ELECTRONIC MUSICIAN on major news-stands. It is creating a revolution in music! Don't let Cromemco be left out of this breakthrough!'

Well, I don't what more I can do to keep Cromemco in the picture other than putting forth a general appeal to the community of Cromemco users: Have anyone of you developed such an interface or would anyone else be interested in developing a MIDI interface for Cromemco systems? Having reviewed the pertinent articles in BYTE, which went so far as to show how to design both the card interface and recording software (source code provided in 'C') for an IBM clone, it appears that producing such an interface for a S-100 based Cromemco system would not pose much of a problem (the interface for a C-10 is a different story). The question is, are there any hobbyists or entrepreneurs out there willing to undertake the project? If so, please contact I/O NEWS or Mr. Burkett. His address is:

Jack E. Burkett
P.O. Box 20788
Houston, TX 77225

XXU Prominent in S-100 Journal

The Fall edition of *S-100 Journal* featured the XXU on its front cover, as well as the article by Ed Lupin which appeared in the last issue of *I/O NEWS*. Copies of *S-100 Journal* are available for $4.90 each ($6.90 outside of the U.S.) and may be ordered from:

S-100 Journal
P.O. Box 1914
Orem, Utah 84057

New Cromemco User BBS

Kim Dildine, of Dildine Industries, Inc., writes:

One day, not long ago, while I was pondering the meaning of life, it occurred to me there was no central depository of Cromemco information. There was no system set up anywhere for people to call up and share their problems or ideas. I figured the Cromemco users of the world should have access to a bulletin board type of system.

I happen to have a rack full of eight-bit System Threes sitting around doing little more than looking impressive. I decided to configure one of them into a Z-80 Cromix system. The computer's only duty will be to serve the needs of those who dial it up.

Until I start getting a lot of people using the system I won't want to spend much time or money to set it up. The system will only be available during non-business hours, which are 9am to 5pm (central time) Monday thru Friday and 10am to 2pm on Saturday. The phone number is 219-931-0203. If you dial this number outside of those hours you should get in. You can use a 300 or 12000 baud modem. Tap return several times until you get the LOGIN prompt. Log in as NEWUSER to get more information. Log in as BBS to use the system.

Once logged in you will be a standard non-privileged user in the Cromix operating system. I hope we don't get a jerk who likes deleting things because there won't be any protection from this kind of attack. The system will be at the mercy of those who use it.

I hope this system will fill a void and that it gets used by many people. I expect everybody to contribute their information, thoughts and programs to make this system useful for everyone.

Give it a try. Let me know what you think.

Kim Dildine
P.O. Box 4189
Hammond, IN 46324
(219) 931-7020 (312) 891-1064

UNIX Plotters?

Doug Sharrod, of Multi-Media Video, is interested in connecting a graphics plotter to a CS-300 running Unix System V. He would like to hear from anyone who has done this, or knows how to do it. If you have any information, please contact Doug at (408) 727-1733.

68000 Chess Program?

IACU member Chuck Montgomery, of Pennsylvania, is going to be upgrading his system to Cromix-Plus, series 40, running with a XXU, and is interested to know if anyone has a chess program to run under the 68020-based XXU. There are numerous chess game programs available in the public domain, and Chuck was wondering whether or not any have been ported to series 40 Cromix-Plus. If you can help in this regard, please contact *I/O NEWS* or Chuck Montgomery. Chuck can be reached at (412) 458-7500 ext. -17.

Computer Books

If you are interested in hearing about the latest publications about computers, operating systems, programming, etc., here's a hot tip: get on the *COMPUTER LITERACY BOOKSHOP* mailing list! You'll receive, on a monthly basis, their *NEW BOOK BULLETIN* — an eight page newsletter which gives abstracts and pricing information on all of the latest computer-related books. In addition, you can request Prentice-Hall's *C/UNIX COLLECTION* which describes 28 distinctive guides to the UNIX operating system and C programming. To get on their mailing list simply write to:

COMPUTER LITERACY BOOKSHOP
520 Lawrence Expressway
Sunnyvale, CA 94086
(408) 730-9955

System 120

Continued from page 7

TABLE

MODEL	DISK STORAGE		RAM MEMORY		OP. SYSTEM		PRICE
	50 MEG	140 MEG	2 MEG	4 MEG	CROMIX +	UNIX	
CS120H50XXC20	X		X		X		$14,995
CS120H150XXC20		X	X		X		16,995
CS120H50XX20	X		X		X	X	16,995
CS120H150XX20		X	X		X	X	18,995
CS120H50XX40	X			X	X		16,995
CS120H150XXC40		X		X	X		18,995
CS120H50XX40	X			X	X	X	18,995
CS120H150XX40		X		X	X	X	20,995

For Sale - Complete CROMIX System

Complete Cromemco 3 User System under Cromix. Consisting of Z 2D, HDD 10 meg. Hard Disk, Control Data 35 meg Hard Disk, two 5¼" and two 8" Floppies. One 3102 terminal and a C10 64k stand alone computer. Used as a terminal for cromix or a stand alone computer with its own 5¼" floppy.

Boards include: ZPU, PRI, 16FDC, 2 Tu-art, 4 64k Memory, and never used IOP and Quardart. Plus one extra ZPU.

Software includes: Cromix, Gunn Ent. Enhanced Cromix with CPM Simulator, DBase II, Fortran IV, Writemaster, Planmaster, Structured Basic, and Electric Blackboard Multi-window text editor.

Take it all for $2000

Call Mike Mellone at (201) 479-6134

NEW PRODUCTS...

NEW PRODUCTS is a regularly appearing column devoted to announcing and following hardware and software products of interest to Cromemco users. Most information is derived from press releases submitted by vendors. As a result, I/O NEWS cannot be responsible for errors of ommission or any other inaccuracies.

Safe C

Catalytix Corporation of Cambridge, MA, the leading supplier of advanced software development tools for C programmers, announced today the availability of its entire **Safe C™** product line for Cromemco's UNIX-based computers. Cromemco, of Mountain View, CA, manufactures multiuser computer systems based on Motorola's 68010 or 68020 microprocessors running UNIX System V.

Catalytix' Safe C family of software development tools consists of an advanced **Interpreter** for interactive execution of C programs and **Runtime Analyzer** for automatic detection of C programming errors. The Analyzer also includes modules which perform tracing, facilitate code optimization, and analyze test coverage. These products are useful for developing new software applications and porting existing C software to Cromemco's machines.

Catalytix has also released a special version the **C Trainer** for Cromemco's UNIX-based microcomputers. The C Trainer is an interactive instructional package for teaching the C language. It includes a textbook published by Prentice-Hall, *The C Trainer Interpreter*, and an on-line library of programs described in the textbook.

Current users of the Safe C family include AT&T Bell Laboratories, IBM, Harvard University, Hewlett-Packard, Prime Computers, Lawrence Livermore National Laboratories, Commercial Union Insurance, Raytheon Company, General Electric, U.S. Department of Defense, Sandia National Laboratories, and the University of California.

Prices for Catalytix software for Cromemco systems are: $2,000 for the Interpreter; $1,200 per Runtime Analyzer module; and $220 for the C Trainer Package. A dial-up demonstration of Catalytix Safe C Products is available to those with a 1200 baud modem. For additional information contact:

Jean M. Kelly
Catalytix Corporation
55 Wheeler Street
Cambridge, MA 02138
(617) 497-2160

Safe C is a trademark of Catalytix Corporation.

TODAY

Cromemco has announced the availability of **BBJ's TODAY fourth generation language (4GL)** for its line of UNIX-based microcomputers. With the TODAY 4GL applications can be developed in a fraction of the time required with classical programming languages such as BASIC, COBOL, etc., and easily adapted and modified to changing needs. The TODAY 4GL is a proven system. The product was originally developed by BBJ for Hewlett-Packard computers, and has been extensively used with Cromemco computers for over two years. One Cromemco user is Algorithm, Ltd., the exclusive Cromemco distributor in Greece. Mr. George Natis, Algorithm's customer support manager, writes:

"After TODAY 4GL, everything else is history. An application that took us 3 months to write with C and Unify took just 3 weeks with TODAY. We are running nine users on a CS400 with 40 megabyte C-ISAM files with an extremely fast response time. We are very enthusiastic about TODAY 4GL and Cromemco."

Some of TODAY's key features are:

- **Simple to develop sophisticated applications:** TODAY anticipates the needs of the developer. Without procedural programming, TODAY can create menus, on-line transactions, reports, background jobs, on-line help information, error messages, application documentation and complete data base definition for your requirements.
- **Hides the operating system:** User logs in directly to the application and never sees bin files or directories.
- **Fast development:** Little or no programming is involved. Menus, prompts, help messages, and screen "painting" capability all aid rapid development.
- **Foreign language support:** The use of synonyms allows easy language conversion.
- **Highly efficient execution:** TODAY is written in the 'C' language; development under TODAY creates a P-code. An Exerciser has been ported to Cromemco which is optimized to run the P-code taking advantage of Cromemco speed and features. The Exerciser requires only 250kb of memory with 30kb for each user.
- **Database interface:** TODAY interfaces to C-ISAM, Informix/SQL, and UNIFY.
- **Documentation and maintenance:** Automatic audit trail and documentation features relieve the developer of much of this time-consuming chore. Maintenance becomes easier.
- **Training:** BBJ provides a one week training course in California and Florida.

The TODAY 4GL can be ordered directly from Cromemco (model TODAY-XS and a run-time only version, TODAYRT-XS). For further information contact Cromemco or:

BBJ Computer Services Inc.
2946 Scott Blvd.
Santa Clara, CA 95054
(408) 727-4464 Telex: 5101012118

Editor's Note:
We look forward to running articles and reviews on this exciting new product in upcoming issues of I/O NEWS.

Telephone Systems Software

Computer Crossroads has written a package in TODAY 4GL that gathers call information from one or more ROLM "CBX" phone systems and prepares management reports using this information. The program, called **TCAS**, generates reports including time of the call, phone extension, number called, and length of call for both incoming and outgoing calls. The software runs on a Cromemco CS100H50X10 or larger computer and requires the TODAYRT-XS run-time software. TCAS is available from Computer Crossroads. (It is not available directly from Cromemco.) List price for the software is $5,000. Address inquiries to:

Computer Crossroads of America
1750 Alma Road #118
P.O. Box 832117
Richardson, TX 75083-2117
(214) 231-6108

2nd Generations FERRUPS UPS

Ten "Second Generation" FERRUPS Uninterruptible Power Supply (UPS) models featuring the best price/performance ratio in the industry are available from **BEST POWER**

FOR SALE

Cromemco System II with 68000 DPU, Cromix 20.65, (4) 512K memory boards (no ecc), (2) IOP's & Quadart, 64FDC, PRI, FFP, Wren Hard Disk (30Mb), 2 5¼ in. floppies, BRZ-II, Cipher 9-track tape & controller, C-10 with upgraded keyboard (CKBC).

All standard software & Fast Fortran '77. Well maintained (under contract).

Asking $13,900 or Best Offer.

Will provide free Tektronix graphics driver to purchaser and have system certified if desired.

Respond to:

P.O. Box 153
Fulton, MD 20759
or call Bill at (301) 730-0685

WANTED IOP/QUADART

Exchange or sell 3102, 3100 terminals, System III, HDD-11, BRZIII in Cromemco desk.

Bill Beeman
(613) 225-4008

excalibur

UPGRADE YOUR SYSTEM!!

256KZ to 1024KZ	Qty 1	$ 795
	2-4	595
	5-10	475
	11 +	400
512MSU to 2048MSU	Qty 1	$1,695
	2-4	1,270
	5-10	1,020
	11 +	900
Excalibur Utilities	Qty 1 +	$ 99

90 day warranty on all modified boards.

Need to upgrade your memory but can't afford the downtime?
Call and ask about our **ADVANCE BOARD PROGRAM!**

EXCALIBUR COMPUTERS
4548 Auburn Blvd., Suite 191
Sacramento, California 95841
(916)971-9610

TECHNOLOGY, INC.

These models feature sophisticated microprocessor controls, high efficiency and decorative, space-saving design. BEST'S new units range in capacity from 15KVA to provide minicomputer
microcomputer applications. All units provide true on-line uninterruptible power that protects against all line power problems including power blackouts, brownouts, sags, spikes, surges, noise and frequency variation.

According to Terrance D. Paul, President, "The Second Generation line, employing our proven ferroresonant technology, presents the most reliable and efficient UPS at the lowest price available today." BEST is one of the three leading suppliers of UPS in its capacity range.

Among the features of BEST's Second Generation products are:

• **Lower Cost**. BEST'S prices are as much as 50% below competitive units. A BEST 3KVA unit, for instance, lists at $5,995, while comparable units from its two major competitors are prices at $14,885 and $12,909.

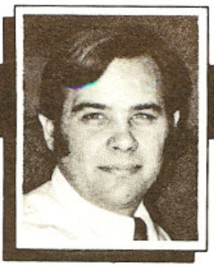
• **High Efficiency**. Since BEST's technology eliminates the AC to DC to AC double conversion typical of other UPS, 90% of the incoming line power is passed on to the load, instead of 60 to 80% in competitive units.

• **Improved Reliability**. Because the inverter and batteries are only "on" when needed, the life of these systems is extended dramatically.

• **Microprocessor Control**. The 2 thru 15KVA units have a keyboard to control the microprocessor and a LCD display of information. Immediately available to the user are 17 meter functions, 12 alarm messages, and 13 set points to tailor a FERRUPS to a specific application. Virtually all set-up, calibration and other user-defined control of the unit is exercised from the keyboard., The microprocessor controls the inverter as well as coordinating input and output systems.

For more information contact:

Cheryl A. Martin
BEST POWER TECHNOLOGY, INC.
P.O. Box 280
Necedah, WI 54646
(800) 356-5794
In Wisconsin: (608)565-7200



TEC TIPS

TEC TIPS is a regular column aimed at providing hints for keeping systems up and running. It will not attempt to deal with specific engineering applications or non-standard configurations. TEC TIPS is edited by Richard Quinn, owner of QUINN TEAM, a Southern California computer service firm. Telephone (818) 889-4819

Abstract:

This is the first of a two-part installment which covers hard disk initialization under Cromix-Plus and the "art" of making cables for peripheral-to-Cromemco interconnection. The use of the Cromix-Plus utility inithard is examined. A history of Cromemco hard disk subsystems and controllers is presented. Methods for partitioning hard disks for use under Unix or Cromix-Plus are specified. Under the cables topic, the serial and parallel interfaces (Centronics and Qume) employed in Cromemco equipment are discussed. The pin-out table for a Centronics Parallel to Cromemco is provided. Serial interfaces are similarly discussed. The second installment will provide a more in-depth study of the RS-232 serial interface and the principle of hardware-to-software "handshaking" protocols in relation to printers and modems.

STDC Hard Disk Initialization

There still is some confusion on hard disk initializations using inithard under Cromix-Plus, especially where partitions are concerned. Let me see if I can shed some light on the topic, not from a "how to" standpoint, but from a logical standpoint so that you can see what the creators had in mind.

Note: There is a bug in the current release inithard shipped with Cromix- Plus version **31.40**. The inithard version is 30.15. It does not properly write the segment table so the drive always appears to have only one segment, STD0. Use version 30.07 to initialize hard disks the first time until this minor bug is fixed. Using 30.15 to re- initialize an already initialized drive seems fine.

For purposes of this discussion I will use two terms. The first is "logical disk drive" or "logical drive". A logical drive may also be called a segment or partition, and is actually a segment of a physical drive. The second term is "physical drive" which refers to the piece of hardware.

In the case of Cromemco systems this is the WDI type using IMI (International Memories, Inc., now out of business except for service) type hard disk drive originally sold as model numbers HDD-11, HDD-22, HDD-5 and HDD-20 or STDC-type hard disk drive using the industry standard ST-506 type drive originally sold by Cromemco as the HDD-21 and HDD- 50. WDI drives all worked under both CDOS, Cromix-D (the original Cromix for DPU-based 68000 systems, now replaced with Cromix-Plus) and Z80 Cromix. They did not work under the original versions of Cromix-Plus but have been supported since version 30.79 provided you have a WDI-II, not the original WDI. (You can use any version of the IMI hard disk drive if you have a WDI-II controller card.) STDC-type hard disk drives work only on Cromix systems and not on CDOS systems.

Less popular, but still common, are the SMD type hard disk drives that are on some Cromix-D and Cromix-Plus machines. These often have removable platters and are usually more expensive mini-computer-like drives. These drives are supported on the Cromemco SMDI card.

The newer, faster, and now more popular STDC-type hard disk drives are so called because they use the more recent ST-506 interface standard instead of the older IMI interface. Cromemco was the first to offer a hard disk drive as standard equipment and International Memories was the only show in town at the time, hence the "IMI Interface Standard." But the ST-506 standard came about much later and its acceptance in the IBM PC machines has made it dominant. There are several newer standards that currently are contending for the next generation of hard disks that will certainly offer greater storage and performance.

Anyway, back to logical and physical disk drives. The IMI type interface can typically support up to 3 physical drives: 5, 11, or 20 megabytes (megabytes will henceforth be referred to as "megs"). The STDC can support up to two drives, many different types from 5 to 240 megabytes, with the most common in the 20, 30, 35, 50, and 85 meg range. These are physical drives.

Any of these drives can be segmented into one or more (up to 32) logical segments or logical drives. That is, they appear to the computer as a wholly separate disk drive even though all 32 segments may be on the same physical drive. You mount and unmount them like any other hard disk or floppy disk. You initialize them and make file systems on them as though they were separate physical drives. But in reality many different logical drives may all reside on the same physical drive. This feature has ONLY BEEN true on Cromix- Plus and UNIX machines. Prior to these operating systems there was only one physical and one logical drive and one device number.

The obvious advantage of separate logical drives, especially the large capacity drives, is that it gives the user a much more flexible method of organizing data. The segments can be sized by their functions and this will make backing up easier. For instance the operating system occupies about 2 megs and could be put on a segment of its own to keep it separate from other data. Back up this segment only when a change happens. I usually segment the drive in 25 meg or smaller segments, all the same size, so that each will fit on a CTD tape without a break and so the contents of one segment will fit entirely on another segment. That makes restoration easy if problems occur. If you are using floppies to back up, smaller segments may be better. Also if you have extra space on the drive and the segments are about the same size, you can back up one segment on an unused segment.

To allocate the amount of space per segment I use the following simple formula. I take the total capacity of a drive and decide how many segments it will take to keep each within the size limit that I want. Let's say we are working with an 85 meg drive and don't want more than 20 megs on a segment so that we can back up one segment per tape. 85 megs gives about 4 segments at 21 megs each. If the drive has 1224 cylinders (number of heads does not matter for this purpose) then 1224 divided by 4 is about 306 cylinders per segment. So when we have run inithard on the whole drive and it asks if we want to declare partitions or segments we answer "yes" and give the following values: 306, 612, 918 and carriage return to end. The segments will then look like this: cylinders 0-306 = STD0; cylinders 307-612 = STD1; cylinders 613- 918 = STD2; and cylinders 919-1224 = STD3. If you don't declare any partitions, the whole drive will be STD0. Now we have done the original initialization AND segmentation of the PHYSICAL drive setting up the various LOGICAL drive segments.

If you make a file system using makfs on any segment (logical drive, not physical drive) it will treat that segment as a separate drive and will have no effect on the other partitions on the same physical drive. The same is true of inithard. If you initialize only STD1, this will have no effect on the contents of other segments on the drive. Again, if you use segment STD31 this will be the whole drive regardless of segments and WILL wipe out data on ALL logical segments. Only use STD31 when setting up a new drive or changing segments on an existing drive.

Some other rules to remember: When a new drive or badly scrambled drive is initialized it must be initialized as STD31 if it is the first physical drive or STD63 if it is the second physical drive. Remember a physical drive can have up to 32 (0-31 inclusive) total segments, even if you don't use them all. The first physical drive is segments 0-31 and the second 32-63. If the original label and segment table can be read on the drive giving the number of heads, number of cylinders, where write precomp starts, segments, alternate tracks, etc, then these will be the defaults when you re-initialize the whole drive. Otherwise, if you want to change them or restore one that the system cannot read, you must know and enter the proper values.

I ALWAYS print out the disk information using the diskinfo utility BEFORE I have a disk scramble. You can't always get this information after a scramble. I redirect it to a printer and save the information with system documents in case I ever need to re-initialize the disk. For STDC Drive One use:

```
diskinfo std31 > /dev/prt
```

For the STDC Drive Two (if present) use:

```
diskinfo std63 > /dev/prt.
```

The first segment is usually the operating system's root device, normally STD0. This is not necessary but is common practice. Any segment can be the root provided it has all necessary system utilities, directories and files and is declared the root device on booting.

When writing a boot on a drive to allow an automatic boot directly from RDOS (MUST use RDOS version 3.12 for STDC-type hard disk drives and version 3.08 for WDI type drives) you should use the segment that is the root device. Thus if you have only one STDC-type drive you would write a boot to the drive as follows: wboot std0 or in the case of a WDI type hard disk: wboot hd0. This is the method if writing a boot to a hard disk IF you have an RDOS version capable of booting from your drive. Otherwise you must boot from floppy. This is where most people have a problem. Writing a boot does not care how many segments you have or what segment is the root device. RDOS reads the boot off of the whole device referred to as STD31.

Setting the switches on the 16FDC or 64FDC will cause a default boot to this drive or entering bst0 or bh0 at the RDOS prompt will cause a boot. Under all circumstances, all versions of RDOS will do a floppy boot from floppy disk. But using the proper hard disk versions of RDOS will not only allow a faster hard disk boot but will also allow you to run simple tests on a defective drive if you cannot otherwise boot the system.

To summarize:

- Whenever you do anything that effects the whole physical drive refer to it by the highest segment number for that physical device, i.e. the first physical drive is STD31, the second STD63.
- Choose segment sizes that are logical for what you are doing on the system. Creating random size segments on an 85 meg drive or creating no segments makes system administration difficult and limits what you can do to restore a scrambled drive or segment.
- Calculate the segment sizes by taking the total drive capacity divided by the size segments you want and then divide the total cylinders by the number of segments desired. An example on an 85 meg drive: 85 megs total divided by desired 20 megs per segment gives 4 segments total. If the drive has 1224 total tracks that is 306 tracks per segment. If you don't want all segments the same size, as in UNIX, another way to approach the problem is to determine the capacity per cylinder and then the number of cylinders per desired segment size. Example: if I want a 10 meg and two 35 meg segments on the 85 meg drive I could take 85 megs divided by 1224 cylinders for about 69.5 Kilobytes per cylinder. I can then figure the number of cylinders needed for the desired segment size.
- When writing a boot to the drive treat the drive as a physical drive referring to it as STD31 or STD63.
- If you don't need to re-initialize the whole drive but only want to change segments answering "x" to the first cylinder question on inithard will allow you to change the segment table without re-initializing the whole drive. But if segments are changed you will lose data on existing segments and must make new file systems on each re-sized segment. Therefore data should be backed up before changing segment tables or it likely will be lost (obviously this is true when initializing segments too.)
- All device segment names must be in the /dev directory if they are declared. STD31 must be present to initialize the drive in the first place and STD0 to refer to it as a minimum if I declare no partitions. Above these, a device name must be present for each additional segment. For example, if I declare six segments the devices STD0, STD1, STD2, STD3, STD4, STD5 and STD31 must be present. Use makdev to add them to the device directory if they are not already there.
- File systems must be created on each segment as if it were a separate physical drive using makfs. If a file system is not made on a segment the message "Not a Cromix disk" will occur whenever you try to mount the segment.
- Making a file system or initializing a segment will have no effect on other segments with the exception of STD31 or STD63 which refer to the whole drive.
- New drives need initialization information to properly initialize them in the beginning. Drives already initialized can be re-initialized and original values used when the drive was first initialized will be the defaults — assuming the label is intact on the drive (see *I/O NEWS, Vol. V, No. 4 — TEC TIPS* for

PC Network / Cromix INTERFACES

*** 3COM ** NOVELL ***

PERSCI REPAIR

SPECIALISTS IN CROMEMCO/PERSCI
PROFESSIONAL SUPPORT AND DOCUMENTATION
REPAIR SERVICE FROM $200.00
30-DAY WARRANTY ON SERVICE & PARTS

C-10 GRAPHICS

UTILITIES TO DRAW & PLOT
FOR SBASIC AND PLANMASTER
GRAPH-10 $80.00

TREXIS Incorporated

45F ROUTE 303
VALLEY COTTAGE, NY 10989
(914) 268-5161

recommended initialization values).

Cables, Cables and More Cables

It is impossible for me to give you all of the possible cable combinations required for attaching various peripheral devices to a Cromemco. There are so many possibilities and combinations, sometimes in the same pieces of equipment. But let me offer a few short pieces of information and principles that I use when setting up hardware.

First there are two basic peripheral-to- computer interfaces: parallel and serial. Within each there are several variations. Let's start with parallel.

Cromemco uses two parallel interfaces: the Qume interface and the Centronics parallel interface. The Qume interface was developed early on by the Qume printer company and was the industry standard for high quality letter quality printers. It is ONLY supported on the Cromemco PRI card, i.e., no C-10s. Currently it is not supported under UNIX but is under CDOS and all versions of Cromix. It is a 16-bit interface and allows for truly proportional spacing and has a great deal of flexibility, using mode to set line and character spacing. It uses the "tip" device driver. The printers sold by Cromemco for use with this drive were the 3355 (Qume built daisy wheel), the 3355A (built by NEC and sold as the NEC 7700, 7703, and 7700Q) and the 3355B (an updated version of the 3355A also built by NEC). Software-wise these printers were virtually interchangeable. Each has its own cable and uses an extensive 25 conductor cable good up to a distance of 25 feet from the computer. This is the best word processing printer Cromemco has sold.

The next parallel interface is the Centronics interface: a 7 or 8 bit interface based on the now defacto standard Centronics. It is a simpler interface and transfers all eight data bits in one cycle and has a hard-wire from the computer to the printer for each bit and control line. There are usually at least 11 wires: bits 1-7, busy line, strobe, ready line, and ground. Cromemco Centronics uses only 7 bits with the exception of the C-10 which can do 8 bits. The 8th bit is usually only used for graphics on some printers and is tied to ground when connected to a Cromemco to keep it low. Use the following connections when making a Centronics cable.

Centronics Parallel to Cromemco in Centronics Order

Cent 34 pin	Signal	DB-25	Color
1	Strobe	22	Black
2	Data 0	13	Orange
3	Data 1	25	Gray
4	Data 2	12	Red
5	Data 3	24	Tan
6	Data 4	11	Yellow
7	Data 5	23	White
8	Data 6	10	Brown
9	Data 7	14	Pink (tie w/Green)
10	Ack.	15	Violet
11	Busy/Ready	17	Blue
14,16,21-23	Ground	14	Green

The cable can be up to 50 feet long and longer if shielded cable is used. Care should be taken as solder joints are close in the connectors and bridges can occur easily. The colors listed are only my order and have no other significance.

Serial cables are usually easier to make but harder to figure out as they are not all the same, as with the Centronics cables. The basic serial interface is used between virtually all terminals and computers and now more than ever is used between computers and printers. It can cover a much greater distance and uses only a few wires instead of the 12 to 25 wires used with parallel interfaces. The basic cable has a transmit line, a receive line, and a ground line. But now the problem.

A transmit line may not always be a transmit line. Not that there is any difference in a transmit line, but remember that a transmit line must be hooked to a receive line. Therein lies the difficulty. The pin number that is transmit on a terminal is not the same pin number on the computer.

On the Cromemco all types of serial interfaces have the same pin numbers, so that helps. Whether it is an FDC, TUART, or OCTART, the pin numbers are always the same. The QUADART is the sole exception. It has two sets of pin connectors on the card for each port and one is just the opposite of the other.

For all interfaces except the QUADART use the following rule of thumb. When I say "transmit" that is from the standpoint of the computer. Obviously the computer's transmit line must go to the terminals receive line. If the pin number is not listed, it doesn't count.

Pin Number	Description	Signal Direction
2	Receive Data	From the terminal to computer
3	Transmit Data	From the computer to terminal
7	Data ground	Signal Ground, common

On all serial interfaces these are the basic signals that are required. For CRTs you must have a transmit and receive line for the terminal to talk back and forth. Most terminals do not require any other lines for proper operation. But most terminals, including modems and printers, do have several other lines (mostly on newer pieces of equipment), and these lines are not always needed.

These lines are part of the so called RS-232 standard. The lines that are present other than transmit, receive, and ground are for hardware handshaking (handshaking is a general term meaning coordinating hardware/software so one keeps up with the other).

In the next *TEC TIPS* column I will try to cover these lines, their uses and what to do with them. I'll cover them especially in relationship to printers and modems.

JORDAN SIEDBAND

Computer System Solutions
Specializing and Recommending

Cromemco Systems

5017 Fairview Lane
Skokie, IL 60077

(312) 674-1175

THE HACKER'S HOME

THE HACKER'S HOME explores techniques in 'C' programming for Cromix and UNIX. Users are encouraged to submit utility programs of their own. It is edited by Rick Dhaenens, Senior Manager for the Cromemco, Inc. Atlanta Regional Office, 5901-C Peachtree Dunwoody Rd, Suite 375, Atlanta, GA 30328. (404) 391-9433.

Termlib

Since I have access to both Cromix and Unix systems, most of the utilities I write in 'C' are used on both operating systems. This leads to several problems that need to be solved for both operating systems. Terminal independence is also required to do the screen I/O for most of the utilities. To solve these problems I have developed a library of routines that are commonly used but must be implemented differently on Cromix and Unix.

These routines cover the basic termcap and I/O control functions needed to write interactive programs. Termcap facilities are available in Unix System 5 and in the Cromix-Plus operating systems. Basic I/O control to set the binary or echo mode and determine if a character is ready can also be implemented on both systems.

Termcap is a description of TERMinal CAPabilities. These capabilities are usually related to the video attributes supported by the terminal. Termcap entries in the file /etc/termcap under Unix or /etc/termcaps under Cromix are the character sequences needed to set the terminal to the specified mode. For example the term 'so' should be the sequence to set the terminal to start the standout mode (inverse) and the term 'se' should be the sequence to end the standout mode. These descriptions are read and manipulated by calls in the libtermcap.a file under Unix or syslib.obj under Cromix. Both systems will get the correct termcap entry based on the environment variable TERM.

To use this library in a program there is only one rule you must follow. You must call the routine t_init() before using any of the other routines in the library and call the routine t_end() before exiting back to the operating system.

The test program termtest.c shows how to use these routines to paint the screen then wait for a character to be typed. If the line that says it is in inverse is not then the 'so' and 'se' capabilities are not in the termcap entry for your terminal. Likewise the underscored line needs the 'us' and 'ue' capabilities. More information on termcap can be found in the termcap(5) manual entry on Unix or the termcaps manual entry on Cromix-Plus.

The routine t_init() takes care of saving the modes of the terminal port and initializing the termcap strings. All other routines in the library use these variables to perform their functions. These variables can also be used in your application program. The initialization sequence 'is', the visual mode initialization sequence 'vs', and the cursor addressing mode initialization sequence 'ti' are sent to the terminal.

The t_end() routine will restore all modes to the modes in effect when t_init() was called. The exit visual mode sequence 've' and the exit cursor addressing mode sequence 'te.' are sent to the terminal.

The routines t_setecho() and t_resecho() will turn on or off the immediate echo of characters that are typed. This is useful when you want to accept a character for input but do not want the character to show on the screen.

The routines t_setbin() and t_resbin() will change the terminal modes not to do input or output processing on characters passing through the driver. This also means that characters are processed one at a time, therefore allowing the retrieval of a single character of input.

The t_inready() routine allows you to determine if a character has been typed before trying to get it. This could be used to check if a key has been typed before continuing execution of a loop.

The t_getnum(), t_getstr(), and t_print() functions are used by other termlib functions but can also be used to retrieve capabilities that are not in termlib.

The t_goto(line,column) function positions the cursor to the specified line and column number.

The rest of the termcap functions are pretty much self explanatory:

t_insl()	insert line
t_dell()	delete line
t_clear()	clear screen
t_cleol()	clear to end of line
t_cleos()	clear to end of screen
t_home()	home the cursor
t_curu()	move the cursor up
t_curd()	move the cursor down
t_curl()	move the cursor left
t_curr()	move the cursor right
t_sinv()	start inverse
t_einv()	end inverse
t_sund()	start underscore
t_eund()	end underscore

To compile this library for Cromix-Plus comment out the define for UNIX and REL3 and leave the one for CROMIX, then compile with SVS 'C':

```
c termlib.c; code termlib.i
```

This will result in the file termlib.obj which can be linked to a program like this:

```
crolinker prog termlib /usr/lib/syslib /usr/lib/clib /usr/lib/paslib
```

Note that the syslib library is included to provide termcap functions for Cromix-Plus.

To compile this library for Unix with SVS 'C' comment out the define for CROMIX and leave the one for UNIX. If you have the SVS release 2 'C' compiler comment out the REL3 define as well. Compile with the SVS compiler like this:

```
c termlib.c jcode termlib.i jlinker termlib rm termlib.obj
```

This will result in the file termlib.o which can be linked to a program like this:

```
cc -o prog prog.o termlib.o -ltermcap
```

To compile this library for Unix with UPST 'C' comment out the CROMIX define and leave the defines for both UNIX and REL3, then compile (e.g., cc -c termlib.c). This will result in the file termlib.o which can be linked to a program like this:

```
cc -o prog prog.o termlib.o -ltermcap
```

These basic utilities can also be extended to other operating environments to provide a standard method of accessing these kinds of primitives from your own 'C' utilities.

Editor's Note:

The source code and object code of the programs given in this column are available at no charge from I/O NEWS. Send us a 5 ¼ inch disk or contact the office to arrange for the transfer *of the files by modem.*

```
/***********************************************************************/
/*                                                                     */
/*      System independent I/O routines For the Cromix+ or Unix        */
/*      operating systems using termcap capabilities.                  */
/*                                                                     */
/*      Written by: Rick Dhaenens                                      */
/*                                                                     */
/***********************************************************************/

/*
#define UNIX     1                      /* Define for Unix.             */
/*
#define REL3     1                      /* Define if ATT or SVS Rel 3 "C"   */

#define CROMIX   1                      /* Define for Cromix.           */

#define ERR      -1

#ifdef  UNIX                            /* Is it Unix?                  */
#  include <sys/termio.h>               /* Include termio definitions.  */
#  include <sys/ioctl.h>                /* Include ioctl definitions.   */
   static struct termio t_mode;         /* Define storage for initial mode. */
   static struct termio t_cmode;        /* Define storage for current mode. */
#  define STDIN  0
#  ifdef REL3                           /* Is it AT&T C or REL3?        */
#    include <stdio.h>                  /* Yes then use AT&T STDIO file. */
#  else                                 /* No then it must be SVS C REL2. */
#    include <svsstdio.h>               /* Include SVS stdio file.      */
#    define ioctl     _ioctl            /* Define C functions for SVS.  */
#    define tgoto     _tgoto
#    define getenv    _getenv
#    define tgetent   _tgetent
#    define tgetnum   _tgetnum
#    define tgetstr   _tgetstr
#    define tgetflag _tgetflag
#  endif
#endif

#ifdef CROMIX
# include <stdio.h>
# include <jsysequ.h>                   /* Include system call equates.  */
# include <modeequ.h>                   /* Include mode information.     */
  static int  t_model, t_cmodel;        /* Define storage for initial and */
  static int  t_mode2, t_cmode2;        /* current modes.                */
  static int  t_mode3, t_cmode3;
#endif
                                        /* Define global variables.      */
int     no_col,  no_lin;
char    *lf_key, *rt_key, *bs_key, *hm_key, *up_key, *dn_key;
char    *init_s, *vint_s, *vend_s, *tint_s, *tend_s, *goto_s;
char    *delc_s, *dell_s, *ceos_s, *ceol_s, *cler_s, *home_s;
char    *rigt_s, *left_s, *curu_s, *curd_s, *insl_s, *insc_s;
char    *inso_s, *eins_s, *undo_s, *eund_s, *invo_s, *einv_s;
char    term_entry[1024], t_buffer[1024],   *t_bp = t_buffer;

char   *tgoto();                        /* Declare non integer functions. */
char   *getenv();
char   *tgetstr();
char   *t_getstr();

/***********************************************************************/

t_init()                                /* Initialize everything, and save */
{                                       /* incoming modes.               */
   char *name;
   int  error, tcchan;

#ifdef UNIX
   ioctl(STDIN, TCGETA, &t_mode);       /* Get the initial terminal modes. */
   ioctl(STDIN, TCGETA, &t_cmode);      /* Get the current terminal modes. */

   if ((name = getenv("TERM")) == 0) {  /* Get the terminal name from env. */
      fprintf(stderr,"Environment variable 'TERM' not found.\n");
      return(ERR);
   }

   if ((error = tgetent(term_entry,name)) == ERR) {      /* Get the termcap */
      fprintf(stderr,"Termcap file not found.\n");       /* entry.          */
      return(ERR);
   } else

   if (error == 0) {
      fprintf(stderr,"Terminal type '%s' not found.\n",name);
      return(ERR);
   } else

   if (error != 1) {
      fprintf(stderr,"Unknown error %d.\n",error);
      return(ERR);
   }
#endif

#ifdef CROMIX
   t_model = t_cmodel = getmode(STDIN,MD_MODE1);         /* Get the original */
   t_mode2 = t_cmode2 = getmode(STDIN,MD_MODE2);         /* Cromix modes.    */
   t_mode3 = t_cmode3 = getmode(STDIN,MD_MODE3);

   if ((tcchan = open("/etc/termcaps", 0)) == ERR) {     /* Open the file    */
      fprintf(stderr, "Can't open '/etc/termcaps'\n");   /* /etc/termcaps    */
      exit(ERR);
   }

   if (tgread(tcchan, term_entry, 1024)) {               /* Get the terminal */
      fprintf(stderr, "Can't get termcap entry\n");      /* entry.           */
      exit(ERR);
   }

   close(tcchan);                       /* Close the /etc/termcaps file.  */
#endif

   no_col = t_getnum("co");             /* Get the numeric variables of   */
   no_lin = t_getnum("li");             /* interest.                      */

   lf_key = t_getstr("kl",&t_bp);       /* Get the character sequence sent */
   rt_key = t_getstr("kr",&t_bp);       /* by the cursor motion keys.     */
   up_key = t_getstr("ku",&t_bp);
   dn_key = t_getstr("kd",&t_bp);
   bs_key = t_getstr("kb",&t_bp);
   hm_key = t_getstr("kh",&t_bp);

   init_s = t_getstr("is",&t_bp);       /* Get the character sequence for  */
   vint_s = t_getstr("vs",&t_bp);       /* the useful terminal attributes. */
   vend_s = t_getstr("ve",&t_bp);
   tint_s = t_getstr("ti",&t_bp);       /* the useful terminal attributes. */
   tend_s = t_getstr("te",&t_bp);
   ceos_s = t_getstr("cd",&t_bp);
   ceol_s = t_getstr("ce",&t_bp);
   cler_s = t_getstr("cl",&t_bp);
   rigt_s = t_getstr("nd",&t_bp);
   delc_s = t_getstr("dc",&t_bp);
   dell_s = t_getstr("dl",&t_bp);
   undo_s = t_getstr("us",&t_bp);
   eund_s = t_getstr("ue",&t_bp);
   invo_s = t_getstr("so",&t_bp);
   einv_s = t_getstr("se",&t_bp);
   curu_s = t_getstr("up",&t_bp);
   curd_s = t_getstr("do",&t_bp);
   goto_s = t_getstr("cm",&t_bp);
   home_s = t_getstr("ho",&t_bp);

   t_print(init_s);                     /* Print the initialization seq.   */
   t_print(vint_s);                     /* Print the visual init seq.      */
   t_print(tint_s);                     /* Print the cursor addrs init seq. */
   return(0);
}

/***********************************************************************/

t_getnum(cap)                           /* Return the numeric value of the */
char *cap;                              /* specified capability.          */
{
#ifdef UNIX
   return(tgetnum(cap));
#endif

#ifdef CROMIX
   return(tgnum(term_entry, cap));
#endif
}

/***********************************************************************/

char *t_getstr(cap,buf)                 /* Get terminal capability string. */
char *cap, *buf;                        /* return null pointer if not there.*/
{
   char *cstr;
   int  size;

#ifdef UNIX
   cstr = tgetstr(cap, buf);            /* Get the string pointer.         */
   if (t_strcmp(cstr,"(null)") == 0)    /* Check if it is really there.    */
      return(0);                        /* If not then return zero. Else:  */
#endif

#ifdef CROMIX
   cstr = t_bp;                         /* Get the pointer to the string.  */
   if ((size = tgstr(term_entry, cap, t_bp, 20)) == ERR)   /* Get string.  */
      return(0);                        /* Return 0 if error.              */
   t_bp += ++size;                      /* Increment the buffer pointer.   */
#endif

    return(cstr);                       /* Return pointer to string.       */
}

/***********************************************************************/

t_strcmp(u,v)                           /* Compare two strings return 0 if */
char *u, *v;                            /* they match.                     */
{
   for (; *u == *v; u++, v++)
      if (*u == '\0')
         return(0);
   return(*u - *v);
}

/***********************************************************************/

t_end()                                 /* Reset all modes and restore     */
{                                       /* terminal to original state.     */
#ifdef UNIX
   ioctl(STDIN, TCSETA, &t_mode);       /* Unix uses IOCTL for modes.      */
#endif

#ifdef CROMIX
   setmode(STDIN,MD_MODE1,t_model,-1);  /* Cromix uses SETMODE.            */
   setmode(STDIN,MD_MODE2,t_mode2,-1);
   setmode(STDIN,MD_MODE3,t_mode3,-1);
#endif

   t_print(vend_s);                     /* Print the exit visual mode seq. */
   t_print(tend_s);                     /* Print the exit cursor addr seq. */
}

/***********************************************************************/

t_goto(y,x)                             /* Cursor addressing (line, column) */
int x, y;                               /* using tgoto.                    */
{
   char tmpstr[128];
   char *t;

   if (goto_s) {                        /* Check if capability exists.     */

#ifdef UNIX
      t = tgoto(goto_s, --x, --y);      /* Generate addressing string.     */
      printf("%s",t);                   /* Print it.                       */
#endif

#ifdef CROMIX
      tprint(tmpstr, goto_s, --y, --x); /* Generate addressing string.     */
      printf("%s",tmpstr);              /* Print it.                       */
#endif

      fflush(stdout);                   /* Make sure it is sent.           */
      return(0);
   }
   else {                               /* Return error if not there.      */
      return(ERR);
   }
}

/***********************************************************************/


t_insl()                                /* Print insert line sequence.     */
{ return(t_print(insl_s)); }

t_dell()                                /* Print delete line sequence.     */
{ return(t_print(dell_s)); }

t_cleos()                               /* Print clear to end of screen seq.*/
{ return(t_print(ceos_s)); }

t_cleol()                               /* Print clear to end of line seq. */
{ return(t_print(ceol_s)); }

t_clear()                               /* Print clear screen sequence.    */
{ return(t_print(cler_s)); }

t_home()                                /* Print home cursor sequence.     */
{ return(t_print(home_s)); }

t_right()                               /* Print cursor right sequence.    */
{ return(t_print(rigt_s)); }

t_left()                                /* Print cursor left sequence.     */
{ return(t_print(left_s)); }

t_up()                                  /* Print cursor up sequence.       */
{ return(t_print(curu_s)); }

t_down()                                /* Print cursor down sequence.     */
{ return(t_print(curd_s)); }

t_sinv()
{ return(t_print(invo_s)); }            /* Print inverse on string.        */
```

```
t_einv()
{ return(t_print(einv_s)); }            /* Print inverse off string.         */

t_sund()
{ return(t_print(undo_s)); }            /* Print underline on string.        */

t_eund()
{ return(t_print(eund_s)); }            /* Print underline off string.       */

/*****************************************************************************/

t_print(s)                              /* Print a char sequence if the      */
char *s;                                /* char pointer is not null.         */
{
   if (s) {
      printf("%s",s);                   /* Print the string.                 */
      fflush(stdout);                   /* Make sure it is flushed out.      */
      return(0);
   }
   return(ERR);                         /* Return error if not there.        */
}

/*****************************************************************************/

t_setbin()                              /* Set input mode to binary. This    */
{                                       /* means to return after each char.  */
#ifdef  UNIX


   t_cmode.c_lflag &= ~ICANON;          /* Turn off input processing bit.    */
   t_cmode.c_cc[VMIN] = 1;              /* Set minimum number of chars.      */
   t_cmode.c_cc[VTIME] = 0;             /* Set the timeout value.            */
   ioctl(STDIN, TCSETA, &t_cmode);      /* Set binary mode.                  */
#endif

#ifdef CROMIX
   t_cmode3 |= BINARY;                  /* Set binary mode bit.              */
   setmode(STDIN,MD_MODE3,t_cmode3,-1); /* Set binary mode.                  */
#endif

   return(0);
}

/*****************************************************************************/

t_resbin()                              /* Reset binary mode.                */
{
#ifdef  UNIX
   t_cmode.c_lflag |= ICANON;                    /* Set input proc bit.      */
   t_cmode.c_cc[VMIN]  = t_mode.c_cc[VMIN];      /* Reset min chars.         */
   t_cmode.c_cc[VTIME] = t_mode.c_cc[VTIME];     /* Reset timeout.           */
   ioctl(STDIN, TCSETA, &t_cmode);               /* Reset binary mode.       */
#endif

#ifdef CROMIX
   t_cmode3 &= ~BINARY;                 /* Reset binary mode bit.            */
   setmode(STDIN,MD_MODE3,t_cmode3,-1); /* Reset binary mode.                */
#endif

   return(0);
}

/*****************************************************************************/

t_setecho()                             /* Set mode to echo incoming         */
{                                       /* characters as they are typed.     */
#ifdef  UNIX
   t_cmode.c_lflag |= ECHO;             /* Set the echo bit.                 */
   ioctl(STDIN, TCSETA, &t_cmode);      /* Set echo mode.                    */
#endif

#ifdef CROMIX
   t_cmode1 |= ECHO;                    /* Set the echo bit.                 */
   setmode(STDIN,MD_MODE1,t_cmode1,-1); /* Set echo mode.                    */
#endif

   return(0);
}

/*****************************************************************************/

t_resecho()                             /* Turn character echo off.          */
{
#ifdef  UNIX
   t_cmode.c_lflag &= ~ECHO;            /* Turn off echo bit.                */
   ioctl(STDIN, TCSETA, &t_cmode);      /* Reset echo mode.                  */
#endif

#ifdef CROMIX
   t_cmode1 &= ~ECHO;                   /* Reset the echo bit.               */


   setmode(STDIN,MD_MODE1,t_cmode1,-1); /* Reset echo mode.                  */
#endif

   return(0);
}

/*****************************************************************************/

int t_inready()                         /* Test if input character ready.    */
{                                       /* Return non zero if char ready.    */
   int arg;

#ifdef  UNIX
   ioctl(STDIN, FIONREAD, &arg);        /* Get the number of chars ready.    */
#endif

#ifdef CROMIX
   arg = getmode(STDIN, MD_STATUS);     /* Check if input buffer is empty.   */
#endif

   return(arg);                         /* Return non zero if char ready.    */
}

/*****************************************************************************/


/*
        Termtest: a program to test the termlib library.
*/

#include <stdio.h>

main()
{
   extern int no_col, no_lin;           /* Define external references.       */

   t_init();                            /* Initialize termlib.               */
   t_clear();                           /* Clear the screen.                 */
   t_home();                            /* Home the cursor.                  */
   printf("This terminal has %d lines and %d columns.", no_lin, no_col);
   t_goto(5,5);                         /* Position the cursor.              */
   printf("This is line 5 column 5.");
   t_goto(10,10);                       /* Position the cursor.              */
   t_sinv();                            /* Start inverse video.              */
   printf("This is line 10 column 10 and should be inverse.");
   t_einv();                            /* End inverse video.                */
   t_goto(15,15);                       /* Position the cursor.              */
   t_sund();                            /* Start underline.                  */
```

Create, Read and Write IBM PC MSDOS disks on your Cromemco Unix or Cromix-Plus System

The IBM PCDOS disk format has become the defacto commercial standard for 5.25 inch diskette data transfer for IBM and other computer systems.

Today, with CScopy from Cipher Systems, you can exchange files with any business supporting standard MSDOS or PCDOS disks or the four million IBM PC/XT/AT compatible computers worldwide.

A single CScopy command gives you simple, quick, data exchange with your customers, suppliers and your office or home PC.

CScopy saves time and money.

Forget expensive disk conversion services, phone calls and long, error prone, serial data transfers. CScopy solves the file transfer bottleneck once and for all!

Join the growing list of businesses supporting standard IBM PC disks:

- Accountants
- Print Shops
- Photo-Plotters
- Consultants
- Typographers
- PC Board Shops
- Data Entry Services
- Public Data Banks
- CNC Machine Shops

Order CScopy from Cipher Systems or your dealer.

Cipher Systems

Box 6105, Stanford, CA 94305
(415) 962-8383

Just $125^{00}

Specify Cromix-Plus or Unix version of CScopy. Single machine license. Foreign orders add $15 handling. All payments in US dollars.

Trademarks: CScopy and Cipher Systems: Cipher Systems; Cromix-Plus and CS-400: Cromemco, Inc.

Copyright © 1986 Cipher Systems

```
printf("This is line 15 column 15 and should be underscored.");
t_eund();                               /* End underscore.                   */
t_goto(20,1);                           /* Position the cursor.              */
t_setbin();                             /* Set binary mode.                  */
t_resecho();                            /* Turn off the character echo.      */
printf("Waiting for a key.   ");
while (t_inready()) ;                   /* Wait for a character.             */
printf("You hit the '%c' key", getchar());
t_setecho();                            /* Turn the echo back on.            */
t_resbin();                             /* Reset the binary mode.            */
t_goto(23,1);                           /* Position the cursor.              */
t_end();                                /* Restore all modes.                */
exit(0);                                /* Return to O.S.                    */
```

```
   termtest:       termtest.o termlib.o
                   cc -O -o termtest termtest.o termlib.o -ltermcap

   termtest.o:     termtest.c
                   cc -cg termtest.c

   termlib.o:      termlib.c
                   cc -cg termlib.c
```

Out of Business Inventory Sale

Brand New — Still in Box — Cromemco 300 — 50 MB Hard Disk — 1.2 MB Floppy Disk — 2 MB Error Correcting Ram — Octart Below Wholesale — $11,000

Have another Cromemco 300 that has been Bench tested for about 10 hours. Same as above — Yours for $10,000

1-(800)-874-6391
Ask for Robert Helms

INSIDE CROMIX

INSIDE CROMIX is an open forum on both eight-bit and 16-bit versions of Cromix. The subject matter is directed towards helping Cromix users derive more from their systems. Members' contributions are invited. INSIDE CROMIX is edited by Jordan Siedband, who can be reached at 5017 Fairview Lane, Skokie, IL 60077, (312)674-1175.

If you work with financial institutions, as I usually do, you are involved with calculations that manipulate money. You get percentages for interest calculations, or bond and security prices. Z80 'C' was ideal for those calculations because all arithmetic was performed in BCD which kept the actual digits in storage, to a maximum of 14 digits. These were usually the actual digits that you needed. However, with the advent of 68000 'C', larger programs that compiled faster and ran considerably faster were now feasible. The price paid, among others, was that BCD arithmetic was replaced by binary arithmetic. Now, your own cleverness is needed to keep those totals correct to the penny!!

Those of you using that great data base package, INFORMIX, have heard of the MONEY variable. Instead of relying on storage of dollars and cents in a variable, we store only cents by using

#define CENTS(a) round(100*a + 0.5)

in the define section at the front end of the program. This avoids error, because storing whole numbers in binary double format keeps digits up to about 15 digits correctly. Storing decimals will usually result in the loss of digits towards the end of the 15 digits. The newer versions of INFORMIX perform the rounding correctly. However, there are some programs where I do not have the luxury of the use of INFORMIX routines.

SVS forgot about rounding functions in their library. A short complaint to Egon at Cromemco's Technical support brought forth the following three subroutines which now satisfy my greed for routines that REALLY WORK!! The third is the key to the correct operation of the first two. Floor(x) gets the next lower integral value below or equal to x. Ceil(x) gets the next higher or equal to x. Round(x) = floor(x + .5), or —

define CENTS(a) floor(100*a + 0.5)

```
double floor(x)          /* emulates UNIX floor() function */
double x;
{
    extern double modf();
    double y;

    return(modf(x,&y) < 0.0 ? y-1 :y);
}
double ceil(x)           /* emulates UNIX ceil() function */
double x;
{
    extern double modf();
    double y;

    return(modf(x,&y) > 0.0 ? y+1 :y);
}
        /* modf()  used by ceil() & floor()  from CROMEMCO 7/12/86  */

#define BITSPERBYTE      8
#define BITS(type)       (BITSPERBYTE * (int) sizeof(type))
#define _DEXPLEN         11
#define _HIDDENBIT       1
#define DSIGNIF          (BITS(double) - _DEXPLEN + _HIDDENBIT - 1)
#define MAXPOWTWO        ((double) (1L << BITS(long) -2) * \
                            (1L << DSIGNIF - BITS(long) +1))

double modf(value,iptr)
double value,*iptr;
{
    double absvalue;

    if ((absvalue = value >= 0.0 ? value : -value) >= MAXPOWTWO)
        *iptr = value;
    else
    {
        *iptr =  absvalue + MAXPOWTWO;          /* shift off fraction */
        *iptr -= MAXPOWTWO;                     /* shift back         */
        while (*iptr > absvalue)                /* because of rounding*/
              *iptr -= 1.0;
        if (value < 0.0) *iptr = - *iptr;
    }
    return(value - *iptr);
}

    The responsiblity is yours to store all monies as pennies using the
functions above.  To print, you may have another gift from me,
     printf("%s",format(bucks/100.0));  where format is defined:
```

```
char *format(nmbr)  /* converts  12345.67 to  12,345.67b 16 characters  */
double nmbr:      /* -12345.67 to (12,345.67) (b=' ')    plus '\0' at end  */
{                 /*   J.Siedband          6/15/80                         */
        double x;
        static char cbuf[18];
        char        outbuf[18];
        int i,j,k;

        for (i=0;i<16;i++) cbuf[i]=' ';  /* clear number buffer */
        cbuf[16]='\0';
        x= nmbr<0 ? -nmbr : nmbr;        /* set negative flag   */
```

The responsibility is yours to store all monies as pennies using the functions above. To print, you may have another gift from me:

printf("%s",format(bucks/100.0));
where format is defined:

```
sprintf(outbuf,"%15.2f ",x);      /* note 16 characters  */

for (i=15;i>8;i--) cbuf[i]=outbuf[i]; /* copy righmost 6 chars */
j=i;
while (outbuf[i] != ' ')
{                /* copy chars 3 at a time until exhausted */
        cbuf[j--]=',';
        for (k=0;k<3;k++) cbuf[j--]=outbuf[i--];
}
if (nmbr<0)
{
        cbuf[15]=')';
        i=-1;
        while (cbuf[++i] == ' ');
        cbuf[--i]='(';
}
return(cbuf);
```

Remember that in a previous column, I said that no matter how clever you think you are, there is always someone out there at least as, and possibly, more clever than you. From the first day of accepting this column I have asked you for your contributions and assistance. If my rantings are going to be the only ones here, this column becomes an exercise in self-indulgence. Help us all by sending your ideas to me **now**!

We're Interested!

We're Really Interested...in what you have to say. Especially about how you use your system...the problems encountered and the solutions effected...unusual uses or environments...and any practical applications you would be willing to share with fellow members. These can be short notes for departments like 'bits & bytes...' and 'Tec Tips,' or full feature articles.

Contact Bill Jaenicke at I/O News for editorial guidelines or assistance. We're interested in unleashing your literary talents.

Software

Continued from front cover

I came to a booth in which an Apple II computer sat running an interactive demonstration program. "Press SPACE to continue, ESCAPE to start over, " its screen said. With a mischievous gleam in my eye, I said to my colleague, "Press any other key to crash the system," and pressed a random key. Instantly the screen turned to garbage. Snickering, we hurried away as one of the booth attendants rushed over with a disapproving look on his face. Serves 'em right.

There's a lot of amateur-looking, user-hostile software out there. Several years ago, while manager of Applications Software Development at Cromemco, I developed a list of user-interface principles to be used in developing interactive software. These principles were designed to make interactive software developed in-house at Cromemco easy to use and consistent, and ensure that it was polished and "professional" in appearance. Because I have seen software written outside of Cromemco for Cromemco systems that violates many of those principles, I provide the list (updated slightly) in the hopes that it will improve the quality of the software available for my C-10.

Principle 1 — Consistent Operation

Computer systems should foster the development of habits. When users form habits, they can forget about the system and concentrate on their own work. The more consistent the system is, the easier it is for users to fall into habits.

Within a given program, be consistent in how the program presents information and in how the user interacts with the program. Don't refer to the RETURN key as RET in one place and as RETURN in another. Don't have the ESC key abort a function in one context and RETURN do it in another. Don't require users to type "yes" or "no" followed by RETURN in one situation and simply press "y" or "n" in another.

Strive for consistency between programs. Try to make your notation (e.g., for indicating the default response to prompts) consistent with other programs. Sometimes the emergence of multiple command-entry schemes in a group of programs intended to be used together (e.g., WriteMaster and SpellMaster) is dictated by other considerations, but it makes things difficult for users who must get used to them all.

Principle 2 — Provide clear messages

Messages, error and otherwise, emitted by programs should be neither too terse (e.g., "?") nor too wordy (e.g., "The variable to which you are trying to assign an integer value is of type real"). The old scheme of printing out *** ERROR 47 *** and expecting the user to look-up the error message in a manual is simply unacceptable.

Use real English, not Computerese, when designing messages and when writing documentation. As programmers and engineers, some terms are so much a part of our vocabulary that we forget that they aren't part of Standard English. Some terms to avoid:

'do a compare on X and Y'
Try 'compare X and Y'
'input a number'
Try 'type a number' or 'enter ...'
'run the program X on file Y'
Try 'use the program X with file Y'
'hit RETURN'
Try 'press RETURN'
'SCREEN a file'
Try 'edit a file using SCREEN'
'boot the system'
Try 'load the operating system'
'RAM'
Try 'computer memory'
'when the program comes up
Try 'when the program starts or
'when ... is invoked'
'finish an edit'
Try 'finish editing'

Try to think of some terms that **you** use that might be confusing to someone who has not had much experience with computers. Could these terms be expressed in Standard English?

Don't use the first person in messages displayed by the program. Many engineers and programmers think that having their programs say things like "Hello ... what would you like me to do next?" is cute; that it makes the program "friendlier" to novice users. Actually, the opposite is true. Users don't want their computer to be a friend, a co-worker, or even an assistant. They want it to be a *tool*, and are uncomfortable when it behaves otherwise. If your program is a true Artificial Intelligence system that can do a good job at simulating natural language, O.K.; if not, don't even try.

When designing the prompts and messages that a program will display, think of the context in which the message will appear. By context, I don't mean the state of the program, I mean the state of the interaction.

One good way to do this is to write out protocols (sample dialogs) of a person interacting with a hypothetical program. This method not only helps clarify what the messages and prompts should be, it can even save bytes. If someone asks how old you are, you don't answer, "I am 25 years old." You say, "25."

When WriteMaster was in early stages of development, the message displayed when a user pressed the UNDERLINE WORD key when the cursor wasn't on any word was: 'Sorry...there is no word at the current cursor position." Later, it became clear that, in the context of the *dialog* that has been taking place in this situation, 'Which word?" is sufficient and even preferable.

Principle 3 — Command names should make sense with respect to what they do

Don't require users to learn a new language; use the language that they already know. See the discussion of the UNIX cat command in the article "The Trouble with UNIX" (*Datamation*, November 1981).

Principle 4 — Check the validity of user responses

By far the most common violation of this principle is when a program expects either of two responses, but only checks for one of them, e.g., assumes Yes if the response isn't No, or vice-versa. Check for both!

Assume that users will type invalid responses... they will. In fact, assume that the two-year-old child of the user will pound on the keyboard while her mother or father is in the next room getting coffee. When daddy or mommy comes back, the program should still be there, none the worse for wear. Thus, the program should test for or prevent responses that are too long as well as nonsensical. All too often, programs explode when you simply hold down a repeating key in response to a prompt, or when you press a key that's not on the menu. Design your programs with this idea in mind, and when you're done, sit on the keyboard!

Principle 5 — Allow for error correction; provide a low-risk environment

It goes without saying that a program should allow the user to correct responses as they are being typed... or does it?

Here's the hard one: allow users to correct responses *after they have been entered*. How many times have you typed a response to a prompt, assured yourself that it was what you wanted, pressed RETURN, and then realized that it wasn't what you wanted at all?

Programmers tend to believe that if a user makes this sort of mistake, or types an invalid response, "it's their own fault... they deserve what they get." Wrong: people expect software to protect them from their own mistakes; if it doesn't, they won't buy it, and you'll be out of business.

One way to do this is to have your program offer users the opportunity to correct errors (e.g., Cromemco StatMaster). Another is to allow users to back up or interrupt the normal flow of the program to correct an error made earlier; this is difficult, but not impossible. An excellent way to allow for error correction is to make your application operate like a form: instead of responding to a series of prompts, users move the cursor through different fields in a form, entering data into the various fields in any order they like (e.g., WriteMaster's Set Format command).

Perhaps the best reason for allowing for error correction is that users feel more confident and are more willing to try unfamiliar or complicated features of the system. The result is that users learn to use the software much faster than they do in situations where mistakes are irrevocable, and they enjoy their work more, too.

Principle 6 — Design redundancy into the system

People need, and in fact rely on, redundancy. Don't be afraid of telling the user something twice.

Allow for synonymous commands: different command words that mean the same thing. Just think of all of the times when you couldn't remember the command word to do something, and tried several words that seemed reasonable. Why didn't those other words work? Do you always use the same words for the same things when speaking or writing?

Redundancy allows for error detection and correction. Hardware designers design redundancy into electronic components to increase reliability. The same logic (no pun intended) can be applied to interactive software systems. Why should hardware engineers go to all of the trouble to produce error-detecting memory when the programs that run in it facilitate error by requiring users to codify the universe into nine-digit numbers that contain no extra digits for error detection?

Principle 7 — Design redundancy into the system

People need, and in fact rely on, redundancy. Don't be afraid of telling the user something twice.

Principle 8 — Avoid the video games syndrome: too much information too fast in too many different places at once

The designers of video games have a good reason for having flashy displays that tend to cause information overload: they want users to lose. Designers of business and home applications, on the other hand, want users to win. They want users to feel that *they* are in control.

Principle 9 — Don't impose unnecessary restrictions on users

Many applications programs contain restrictions that seem arbitrary to users. Such restrictions complicate learning and decrease user satisfaction with the software. Many such restrictions are, frankly, due to laziness on the part of programmers who fail to realize how many user-hours one extra hour spent overcoming · a restriction can save. Don't expect users to understand that they can't use files that are larger than 32,768 bytes, or that they can't underline more than five words on one line, or that they have to type in upper case.

Principle 10 — Provide implicit menus if explicit menus are not advisable

For many years, a controversy has been raging in the business computer industry: which is better... menu-driven or command-driven interactive software? Menu-driven programs are usually easier for beginners to learn to use, but the menus only get in the way of experienced users. Command-driven programs are typically difficult to use at first, but more efficient in the long run.

So which *is* better? The answer depends on three very simple factors: 1) how many commands there are, 2) the rate at which a user has to select new commands, and 3) how much *other* information the program has to display. When there are relatively few commands, such that the menu can be displayed constantly, there is no excuse not to do so: it will help new users and it won't get in the way of experienced ones (e.g., the SCREEN and CE editors).

When the number of commands is large enough that either a significant portion of the screen must be sacrificed to the menu or a significant amount of time must be sacrificed to allow for alternating displays of menus and the data being manipulated, the second and third factors come into consideration. If commands are executed often (more than one every 30 seconds) and the screen is needed to display other information, use an implicit menu scheme, which looks like a command scheme but has a menu "hiding behind" every prompt. This implicit menu should not only be available to users when they ask for it (e.g., by pressing the HELP key) or make an error, it should be used by the program to help interpret what they type (e.g., WriteMaster).

If, on the other hand, commands are executed relatively infrequently, an explicit menu is again called for: it won't get in anyone's way often enough to make a difference. For example, the shell, or interactive interface, of an office operating system might display a large menu of possible activities (word-processing, data-base, electronic mail, instructional programs, games, accounting, inventory, system diagnostics, and several locally-developed applications) when a user first logs on. Since users in this situation typically select an activity only once every half hour or so, there is no excuse for making things tough for newcomers by making this selection purely command-driven, and no need to go to the trouble of setting up an implicit menu (e.g., the C-10 Main Menu).

Finally, if there is not much else to display besides the menu, even though it is large, there is again no excuse not to show one (e.g., the old hard-disk diagnostic HDTEST).

Principle 11 — Don't require users to enter non-numerical data in terms of numerical codes

Applications programs that ignore this principle abound, and are probably one of the main reasons that computers have such a bad reputation among laymen. Why, for example, have conversations like the following become a hallmark of the computer age?

"Hello, San Francisco Food Warehouse, John speaking."

"Hello, I'd like to place an order for the ABC Market."

"Fine. We've just gotten a computer order-processing system, so we can process your order much more quickly now. What would you like?"

"Two hundred bags of wheat flour ..."

"What's the item number?"

"What?"

"What is the item number for wheat flour? I have to give the computer an item number."

"I don't know."

"It should be on your price list, right next to the price."

"There's no item number here. Doesn't your computer know the item number?"

"You must have an old price list. I'll mail you a new one, but for now, I guess I'll have to go get a price list and look up everything on your order."

"Ahh, technology..."

Even if the grocer had *had* a price list with the item numbers, he might have read one to John incorrectly, and ended up with 200 bags of dog food instead of flour. The string "wheat flour" is just as effective as an item identifier as is a number, and it has the advantages of being both meaningful and redundant (i.e., if it were misspelled as "wheat flower", people would know what was meant anyway, and the computer would detect an error rather than confusing it with some other item).

Principle 12 — Don't let program internals out into the user interface

A common fault of amateur software is that internal aspects of the program (e.g., variable names, data type names, internal concepts) are visible to users in the form of commands, menu choices, and other program behavior. Just because a block of text is called a TextRect inside the program doesn't mean that users should mark them via Mark TextRect command or menu choice. Just because a game that allows users to adjust its speed of operation uses their input to set a delay doesn't mean that higher numbers should indicate *slower* operation. Users aren't interested in how your program works; they just want to get their own work done. Take their point of view when designing the user interface, and keep the details of the program to yourself.

Further Reading

Of course, there are many characteristics of good interactive software that are not described in the above principles. Workstations with bitmapped screens, mouse pointers, powerful processors, and large memories enable a host of more sophisticated user-interface techniques such as windows and pop-up menus. However, the above list does cover, fairly specifically, the most serious infractions that occur in software written for C-10 class machines.

For a somewhat more general discussion of some of the same issues, see my article "Guideposts to User-Friendliness" in *Data Management*, October 1982. For an excellent discussion of the evils of program "modes", see "The Smalltalk Environment", by Larry Tesler, in the August 1981 issue of *Byte* magazine.

About the Author:

Jeff Johnson *has a Ph.D. in Cognitive Psychology from Stanford University, was a software engineer and Manager of Applications Software Development at Cromemco from 1978 to early 1984, and since then has been working at Xerox Information Systems Division on document editing systems.* IO

Benchmark

Continued from front cover

1) **Calculation types** are benchmarked in specific categories. Thus, users whose programs tend to be dominant in one of the following categories can better judge the performance increase that the MAXIMIZER may yield.

Calculation of transcendental functions (single and double precision)

Input and output to hard disk files (integer data)

Simple arithmetic operations (integer, single, and double precision)

Relational operations (integer, single, and double precision)

2) **Computational speed** (time) for all of the above classes of tests is measured for one, two, three, and four users simultaneously. This set of tests measures system performance degredation with increasing numbers of users.

Benchmark Programs

As an end user of a Cromemco computer I am not inclined to write molecular structure programs in microcode assembler. Rather, I am more interested in how much faster my type of programs will run in real time under FORTRAN-77 (or an equivalent high level language). Furthermore, there may be ways that I may streamline my programs to be even more efficient in a MAXIMIZER enhanced environment while still being somewhat portable to other systems. Likewise, for those contemplating the purchase of a MAXIMIZER it may be helpful to have some idea as to what extent a MAXIMIZER will benefit their computational needs. The Whetstone Suite is useful (particularly in communications from one expert to another). However, I chose to write benchmark programs that would more closely approximate the actual user's applications. This should not be taken as an adverse reflection on the Whetstone Suite, rather, my tests answer my specific questions (to some degree).

Included at the end of this article is the suite of ten benchmark programs written for this analysis. The purpose and function type of each is listed in **Table 1**.

System Configuration

The benchmark programs cited above were run on the following system configuration:

Boards

8 MHz DPU
MAXIMIZER
MCU
3—512 MSU's
OCTART
16FDC (modified to handle 8'' drives)
STDC

Terminals

3102
Televideo 925
C-10
MacIntosh

Mass Storage

2—5¼" Cromemco System I floppy drives
2—8" Shugart half height floppy drives
1—50 Mb Micropolis hard disk

The Error Correcting Memory was turned off. I am aware that my performance would be somewhat improved using KZ memory and the 10 MHz XPU. However, my wallet is of finite depth.

Some tests were run with a DPU upgraded to 10 MHz and showed a 25% across the board speed improvement on all tests NOT using the MAXIMIZER. However, disk read and write difficulties precluded the further use of this modified board.

Results

Each benchtest program was compil-

FOR SALE

Cromemco System 3	software:
Z80 CPU	CDOS
64KZ	CP/M
TuArt	Fortran IV
16FDC	BDS C
2 Perscii 8" DS/DD	MuLisp 80
also:	MINCE
PRI	Supercopy II
4FDC	MCALL
32K Bytesaver	SSG NAD, Qsort
8 " floppy disks	AR

Includes all manuals and documentation. Will take the best offer. Call Bill White at (415)961-4837.

FOR SALE

68000/DPU System with Cromix-Plus, C, BSCIC, & Pascal

$2100.00. (Terminal $150)

Call Robert Coats at (919) 544-2443

FINAL CLOSEOUT!

UPGRADE TO

DOUBLE DENSITY

Hundreds of Cromemco owners have installed the FDCX4 Double Density Upgrade Board and doubled their disk storage. You can, too.

It's easy. The FDCX4 plugs right into your 4FDC board and turns it into a full function double density disk controller.

Some features of the FDCX4 are

- Analog PLL separator for reliability
- Write Precompensation. Allows reliable double density operation with all drives (including PerSci s)
- Works with CDOS, CROMIX, and double density CP/M
- Reads, writes and formats 5" and 8" disks in single or double density. Sizes and densities may be mixed
- Installs quickly and easily. Just unplug 2 chips from the 4FDC and plug the FDCX4 into the vacated sockets. No soldering is necessary if you use only 5" drives. If 8" drives are used, one solder joint completes the installation
- Presently in use by Government agencies, Universities, Hospitals, large corporations and small businesses all over the world
- No risk 15 day trial period

CLOSEOUT PRICE — $79

Price is $~~189.00~~ including shipping. We accept MC, VISA, and AMEX cards. C.O.D. is available only in the USA. California residents include 6% sales tax.

To order, or for more information, write or call:

1601 Fulton Avenue Suite 10A
Sacramento CA 95825
(916) 483-0709

jvb electronics

CDOS & CROMIX are TMs of Cromemco. CP/M is a TM of Digital Research

PERSCI

DISK DRIVE MAINTENANCE

AVERAGE 48 HOUR TURN-AROUND

PPS has been providing solutions for PerSci/Cromemco users, user groups, dealers and OEM's for over 5 years.

With over 43 years of combined technical electronics experience, including 12 years on the PerSci payroll, the experts at PPS are uniquely qualified to provide you with the time conscious results that you require for your highly sophisticated disk drives.

Forget the time consuming multiple warranty returns to accomplish what should have been done right the first time!

All disk drives serviced at our lab are tested for a minimum of four hours on a Cromemco CS-3 to insure operational integrity!

On-site services available in the Los Angeles and San Francisco Bay areas. High volume repairs available on-site internationally.

Also, tandon drive repairs and purchase and resale of new and used Cromemco products.

Call **John Bush**, former Supervisor/Lead Technician of Customer Services at PerSci at **(714) 861-6649**

PPS

(aka Peripheral Labs)

PRACTICAL PERIPHERAL SUPPORT
547 GREAT BEND
DIAMOND BAR, CA 91765

ed under Cromemco's 68000 FORTRAN and 68000 FAST FORTRAN with the fast fortran compiled program taking advantage of the MAXIMIZER. Each program was then run using the clock command which measured real time and CPU time. The results of these tests are shown in **Table 1**. The last two columns in **Table 1** contain the factor change and percentage increase in real time processing speed for each program. For example, Benchtest1 ran 6.84 times faster when using the MAXIMIZER. This corresponds to a 584% speed increase. Equivalently, Benchtest1 took 6.84 times longer to run without using the MAXIMIZER.

The results in **Table 1** immediately show that the MAXIMIZER does significantly enhance numeric coprocessing. Furthermore, it seems to have negligible effect on I/O processes. These results are as expected. However, I see nowhere near the seventeen fold speed increase for transcendental function computation stated in the previous **I/O NEWS** review. I don't doubt their results. Rather, one is unlikely to see that kind of speed improvement in a real programming environment unless he or she is inclined to write code in microcode assembler.

Interesting results are present for the relational operations. For integer comparisons the MAXIMIZER seems to have negligible effect. A three to four times speed improvement occurs for real and double precision relational operations. Thus, a relational operation environment which can function in integer mode will find little enhancement using the MAXIMIZER. This result could be significant for those interested in Artificial Intelligence where relational operations have heavy emphasis. On the other hand, who would want to use a Cromemco for Artificial Intelligence when the company seems to have little interest in pursuing that area? As an editorial comment: "Harry, you might want to reconsider AI."

Table 2 shows the results for the multi-user environment. For these tests each user logged onto a separate directory containing a copy of the benchtest programs compiled under 68000 FAST FORTRAN. With each user working from programs in his own directory, the same benchtest program was run simultaneously. For example, when four users were logged on, all users started benchtest1 within 1 second of each other. The real time and CPU time results for each user are listed in **Table 2**. As can be seen, the CPU time varies little for each of one, two, three, or four users. The real time varies linearly with the number of users. In other words when two users run the same program simultaneously, the program takes twice as long to run for each. With three users the real time increase by a factor of three, etc. The MAXIMIZER and Cromix-Plus deserve good marks for this result since it would be quite plausible to have a bottleneck for degradation in multi-user processing.

Some interesting results are also shown in **Table 3**. However, these results should be reflected upon with care before making any sweeping conclusions. This table shows comparisons between function type for each class of benchmark test. For example, benchtest1 and benchtest2 are the same program except that benchtest2 is written for double precision. The MAXIMIZER will process the double precision program 4.12 times SLOWER than the single precision program. Without the MAXIMIZER the double precision version runs only 2.70 times slower— to me a rather surprising result. This shows that the MAXIMIZER (and environmental software) is less efficient in processing double precision transcendental functions relative to non-maximizer results than it is for single precision. Double precision transcendental function computational speed improvement is less than single precision speed improvement using the MAXIMIZER. This is, likewise, reflected in **Table 1** where benchtest1 runs 6.84 times faster on the MAXIMIZER and benchtest2 is 4.48 times faster.

THE BEST OF C-10 SOFTWARE

Applied Environmetrics now distributes **TYPEQUICK**, the best available typing tutor for the C-10. **TYPEQUICK** teaches the fundamentals of touch typing in ten personalized lessons.

Other fine software products supplied by ***Applied Environmetrics*** are the **C-10 FUN DISK** as well as eight volumes of **THE BEST OF PUBLIC DOMAIN SOFTWARE.**

TYPEQUICK $ 69.50 **THE C-10 FUN DISK $100.00**

THE BEST OF PUBLIC DOMAIN SOFTWARE ($25.00 each)

Volume 1 — Games
(Collosal Cave Adventure, Sbasic Startrek, Trade, etc.)
Volume 2 — Communications
(Modem10, SQ — File squeezer ana unsqueezer, etc.)
Volume 3 — Utilities
(Undelete, Catalogue, LU library utility, etc.)
Volume 4 — Assemblers
(8080 and Z80 assemblers and a loader)
Volume 5 — Disassemblers
(Z80 Disassembler and a Tracer)
Volume 6 — Basic
(Tiny Basic, Sbasic Nolist cracker)
Volume 7 — Chess
(Fortran Source and Executable Code)
Volume 8 — Geneaology
(dBASE II program to enable you to trace a family tree)

Source code is supplied on most of the Public Domain Volumes. They can therefore be easily altered for particular applications and offer valuable examples in applications programming. All prices are in US Dollars and include air mail delivery. Further details from ***Applied Environmetrics, (03) 817-2571.***

ORDER FORM for THE BEST OF C-10 SOFTWARE

***Applied Environmetrics* • 118 Gordon Street • Balwyn, Vic. 3103 • Australia**

PUBLIC DOMAIN SOFTWARE is $25.00 per Volume

Please send:

__ copies of Volume 1 (Games)	$ ___	__ copies of Volume 6 (Basic)	$ ___
__ copies of Volume 2 (Communications)	$ ___	__ copies of Volume 7 (Chess)	$ ___
__ copies of Volume 3 (Utilities)	$ ___	__ copies of Volume 8 (Geneaology)	$ ___
__ copies of Volume 4 (Assembler)	$ ___	__ copies of TYPEQUICK @ $69.50	$ ___
__ copies of Volume 5 (Diassembler)	$ ___	__ copies of the FUN DISK @ $100.00	$ ___
		Total Amount Enclosed	$ ___

Name ______________________

Street & No. ______________________

____

City ______________________

Zip Code ______________ Country ______________

Payment to be by check or bank draft in US Dollars drawn on a United States bank, or in Australian Dollars drawn on an Australian bank.

Conclusions

Clearly, the results of these tests are positive for the MAXIMIZER. For the type of programs which I write (double precision number crunching) I am achieving a seven-fold increase in computational speed by using the MAXIMIZER. Furthermore, the MAXIMIZER seems to perform well in a multi-user/multi-tasking environment. My tests did not show a bottleneck occurring with it.

However, I try to avoid I/O unless necessary in my programming techniques. Anyone who is I/O intensive, even if also calculation intensive, may be disappointed. My suggestion — try it before you buy it.

In the near future there are plans to extend this testing protocol to include a comparison with the MAXIMIZER and XPU and with the newly arrived XXU board.

About the Author

Dr. Stephen Huber *is Associate Professor of Physics at Beaver College and Adjunct Associate Professor of Physics and Mathematics at Drexel University. As a recipient of an ONT Research Fellowship he is presently on leave of absence working on laser interferometry for the Navy.*

Comments and suggestions regarding this article are encouraged and may be sent to:

Stephen Huber
Dept. of Chemistry and Physics
Beaver College
Glenside, PA 19038

TABLE 2

Benchtest results for MAXIMIZER in a multiuser environment
(Results are listed for each user)

Benchtest	8MHz DPU with MAXIMIZER		2 Users with MAXIMIZER		3 Users with MAXIMIZER		4 Users with MAXIMIZER	
	Real	CPU	Real	CPU	Real	CPU	Real	CPU
1	69.0	68.1	139.0	68.1	207.0	68.0	276.0	68.0
			139.0	68.1	206.0	68.2	274.0	68.2
					208.0	68.1	274.0	68.1
							276.0	68.0
2	284.0	281.6	567.0	281.6	850.0	281.3	1131.0	281.2
			567.0	281.6	849.0	281.5	1130.0	281.2
					852.0	281.5	1131.0	281.5
							1129.0	282.3
3	45.0	40.4	83.0	40.4	123.0	40.4	165.0	40.7
			82.0	40.4	123.0	40.6	162.0	40.7
					124.0	40.6	164.0	40.5
							167.0	40.9
4	46.0	43.6	91.0	44.0	135.0	43.8	179.0	44.0
			91.0	43.9	133.0	44.0	178.0	44.0
					136.0	44.0	177.0	44.0
							180.0	44.0
5	98.0	96.5	194.0	96.6	292.0	96.4	389.0	96.5
			195.0	96.4	291.0	96.4	387.0	96.5
					293.0	96.4	388.0	96.4
							389.0	96.5
6	182.0	181.2	365.0	181.2	547.0	180.9	729.0	181.1
			365.0	181.1	546.0	181.2	728.0	181.1
					548.0	181.2	728.0	181.2
							729.0	181.0
7	405.0	402.0	810.0	402.0	1213.0	401.6	1618.0	401.9
			809.0	401.7	1213.0	401.7	1617.0	401.7
					1215.0	402.1	1617.0	401.7
							1619.0	401.7
8	31.0	30.8	63.0	30.8	94.0	30.8	126.0	30.8
			62.0	30.9	93.0	30.9	124.0	30.8
					95.0	30.8	124.0	30.9
							126.0	30.8
9	191.0	189.2	381.0	189.2	571.0	189.2	761.0	189.2
			381.0	189.1	570.0	189.1	760.0	189.0
					572.0	189.2	761.0	189.2
							762.0	189.2
10	380.0	378.3	761.0	378.2	1139.0	378.0	1519.0	378.0
			761.0	377.9	1139.0	378.0	1519.0	378.0
					1140.0	378.2	1519.0	378.0
							1520.0	378.0

TABLE 1

Benchtest performance results for MAXIMIZER with 8MHz DPU

Benchtest	Function Type	Purpose	8MHz DPU		8MHz DPU with MAXIMIZER		Speed Increase	
							Factor	% Increase
			Real (sec.)	CPU (sec.)	Real	CPU		
1	Single Precision	Intrinsic Functions	472	468.3	69.0	68.1	6.84x	584%
2	Double Precision	Intrinsic Functions	1273.0	1267.0	284.0	281.6	4.48	348%
3	Integer	Write to File	49.0	47.6	45.0	40.4	1.09	9%
4	Integer	Read from File	48.0	44.8	46.0	43.6	1.04	4%
5	Integer	Arithmetic	404.0	402.2	98.0	96.5	4.12	312%
6	Single Precision	Arithmetic	1315.0	1309.2	182.0	181.2	7.22	622%
7	Double	Arithmetic Precision	2995.0	2981.7	405.0	402.0	7.40	640%
8	Integer	Relational	32.0	30.8	31.0	30.8	1.03	3%
9	Single Precision	Relational	788.0	784.7	191.0	189.2	4.12	312%
10	Double Precision	Relational	1276.0	1269.5	380.0	378.3	3.36	236%

TABLE 3

A Comparison of Computational Speed for Numeric Types in Each Benchtest Class

Benchtest Comparison	Type Comparison	8MHz DPU	8MHz DPU with MAXIMIZER
TRANSCENDENTAL FUNCTIONS:			
1 to 2	Single Precision to Double precision	2.70	4.12
ARITHMETIC OPERATIONS:			
5 to 6	Integer to Single Precision	3.25	1.86
6 to 7	Single Precision to Double Precision	2.28	2.23
5 to 7	Integer to Double Precision	7.41	4.13
RELATIONAL OPERATIONS:			
8 to 9	Integer to Single Precision	24.63	6.16
9 to 10	Single Precision to Double Precision	1.62	1.99
8 to 10	Integer to Double Precision	39.88	12.26

BENCHTEST1.FOR

```
      PROGRAM BTEST1
C        This program tests the speed of execution of the CPU
C        for arithmetic operations.
      Y=1.0
      DO 10 I=1,10000
         ARG=SQRT((1.+Y**2)**(1./Y))
         Z1=TAN(SIN(ARG)+COS(ARG))
         Z2=ALOG(SINH(ARG)+COSH(ARG)+TANH(ARG))
         Z3=ATAN(ARG/10.)+ASIN(ARG/10.)+ACOS(ARG/10.)
         Y=Y+.1
   10 CONTINUE
      PRINT*, ARG,Z1,Z2,Z3
      STOP
      END
```

BENCHTEST2.FOR

```
      PROGRAM BTEST2
C        This program tests the speed of execution of the CPU
C        for arithmetic operations in double precision.
      IMPLICIT DOUBLE PRECISION (A-Z)
      INTEGER I
      Y=1.0
      DO 10 I=1,10000
         ARG=DSQRT((1.+Y**2)**(1./Y))
         Z1=DTAN(DSIN(ARG)+DCOS(ARG))
         Z2=DLOG(DSINH(ARG)+DCOSH(ARG)+DTANH(ARG))
         Z3=DATAN(ARG/10.)+DASIN(ARG/10.)+DACOS(ARG/10.)
         Y=Y+.1
   10 CONTINUE
      PRINT*, ARG,Z1,Z2,Z3
      STOP
      END
```

BENCHTEST3.FOR

```
      PROGRAM BTEST3
C        This program tests the speed at which data can be
C        written to a file.
      INTEGER A(100,100)
      OPEN(2,FILE=´Bench3´,STATUS= new )
      N=100
      DO 10 I=1,N
         DO 20 J=1,N
         A(I,J)=I*J
         WRITE(2,200) A(I,J)
  200    FORMAT(I5)
   20    CONTINUE
   10 CONTINUE
      CLOSE(2)
      STOP
      END
```

BENCHTEST4.FOR

```
      PROGRAM BTEST4
C        This program tests the speed at which data can be
C        read from a disk file.  The file used is that created
C        by program BENCHTEST3.

      INTEGER A(100,100)
      OPEN(2,FILE=´Bench3 ,STATUS=´old´)
      N=100
      DO 10 I=1,N
         DO 20 J=1,N
            READ(2,200) A(I,J)
  200       FORMAT(I5)
   20    CONTINUE
   10 CONTINUE
      CLOSE(2)
      STOP
      END
```

BENCHTEST5.FOR

```
      PROGRAM BTEST5
C        This program tests the speed of execution of the CPU
C        for integer arithmetic operations.
      N=1000000
      J=1
      DO 10 I=1,N
         J=(I+J)/(J+I)+I*J-J*I+I
   10 CONTINUE
      PRINT*, I,J
      STOP
      END
```

BENCHTEST6.FOR

```
      PROGRAM BTEST6
C        This program tests the speed of execution of the CPU
C        for real arithmetic operations.
      N=1000000
      XI=1.
      XJ=1.
      DO 10 I=1,N
         XJ=(XI+XJ)/(XJ+XI)+XI*XJ-XJ*XI+XI
         XI=XI+1.
   10 CONTINUE
      PRINT*, XI,XJ
      STOP
      END
```

benchtest7.for

```
      PROGRAM BTEST7
C        This program tests the speed of execution of the CPU
C        for double precision arithmetic operations.
      DOUBLE PRECISION XI,XJ
      N=1000000
      XI=1.
      XJ=1.
      DO 10 I=1,N
         XJ=(XI+XJ)/(XJ+XI)+XI*XJ-XJ*XI+XI
         XI=XI+1.
   10 CONTINUE
      PRINT*, XI,XJ
      STOP
      END
```

BENCHTEST8.FOR

```
      PROGRAM BTEST8
C        This program tests the speed at which relational
C        operations are executed for integer arguements.
      N=1000000
      J=1
      DO 10 I=1,N
         IF (I.LT.J) J=J-1
         IF (I.GE.J) J=J+1
         IF (I.NE.J) J=1
         IF (I.EQ.J) J=2
         IF (I.LE.J) J=J-1
         IF (I.GT.J) J=J+1
   10 CONTINUE
      PRINT*, I,J
      STOP
      END
```

BENCHTEST9.FOR

```
      PROGRAM BTEST9
C        This program tests the speed at which relational
C        operations are executed for real variables.
      N=1000000
      J=1
      XI=1.
      XJ=1.
      DO 10 I=1,N
         IF (XI.LT.XJ) XJ=XJ-1.
         IF (XI.GE.XJ) XJ=XJ+1.
         IF (XI.NE.XJ) XJ=1.
         IF (XI.EQ.XJ) XJ=2.
         IF (XI.LE.XJ) XJ=XJ-1.
         IF (XI.GT.XJ) XJ=XJ+1.
   10 CONTINUE
      PRINT*, XI,XJ
      STOP
      END
```

BENCHTEST10.FOR

```
      PROGRAM BTES10
C        This program tests the speed at which relational
C        operations are executed for DOUBLE PRECISION variables.
      DOUBLE PRECISION XI,XJ
      N=1000000
      J=1
      XI=1.
      XJ=1.
      DO 10 I=1,N
         IF (XI.LT.XJ) XJ=XJ-1.
         IF (XI.GE.XJ) XJ=XJ+1.
         IF (XI.NE.XJ) XJ=1.
         IF (XI.EQ.XJ) XJ=2.
         IF (XI.LE.XJ) XJ=XJ-1.
         IF (XI.GT.XJ) XJ=XJ+1.
   10 CONTINUE
      PRINT*, XI,XJ
      STOP
      END
```

FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST

PERFORMANCE PLUS

See Wordstar™, dBASEII™, and other Z80 software really fly!

SLAVE PROCESSORS FOR 11.27 or 31.05 CROMIX

8MHZ Z80H 256K

TWO USERS PER SLAVE

Increase Performance Two to Fourteen Times

Much Faster Terminal I/O

$1095.00 includes software license

(Also ideal for fast process control applications)

NEW 32K STRUCTURED BASIC

Formatted input — High speed built-in Sort

Programs load up to FIVE TIMES FASTER

CROMIX (SBASIC.BIN)	$295
CP/M	$195
UPDATE CROMIX VERSION	$95*
UPDATE CDOS VERSION	$95*

* Must be holder of valid Cromemco License

RUN YOUR FAVORITE SBASIC PROGRAMS ON PC/MS-DOS SYSTEMS

SBASIC.PC	$295
SBASIC.PC RUNTIME	$100

SYSTEMS ATLANTA, INC., P.O. BOX 99, LEBANON, GA 30146 **(404) 928-0240**

Wordstar is a trademark of Micropro. dBASEII is a trademark of Ashton-Tate.

•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•

CROMEMCO COMPUTERS: DESIGNED TO MAKE UNIX SYSTEM V EVEN BETTER...

UNIX System V, the new standard in multi-user microcomputer operating systems, gives you high performance features along with the portability and flexibility of a standard.

Cromemco computers can make UNIX System V even better. Because our systems are designed with UNIX in mind. First of all, we offer UNIX System V with Berkeley enhancements. Then, our hardware uses advanced features like 64K of on-board cache memory and our high speed STDC controller to speed up disk operations – very important with UNIX.

More capability and expandability

We have a high-speed, 68000-based CPU that runs at 10 MHz, coupled with a memory manager that uses demand-paging and scatter loading to work *with* UNIX, not for it.

We provide room for expanding RAM to 16 megabytes – with error detection and correction – for running even the most sophisticated and advanced microcomputer programs. And the power to accommodate up to 16 users – all with plenty of memory.

But we give you even more.

A complete solution

We give you a choice in systems: the System 100 series, expandable up to 4 megabytes of RAM, and the System 300 series, expandable to 16 megabytes. A high speed 50 megabyte hard disk drive is standard on the systems. And you can expand the hard disk capacity up to 1200 megabytes using standard SMD drives. You can add floating point processing. High resolution graphics. Video digitizing and imaging. Communications through standard protocols. Mainframe interface.

And software support is here to meet your needs. We offer major programming languages, database management systems, communications software, including SNA architecture, X.25 protocol, and Ethernet; even a program to interface to an IBM PC if you need to. And, of course, access to the broad range of standard UNIX applications programs that is growing dramatically every day.

Easy to use.

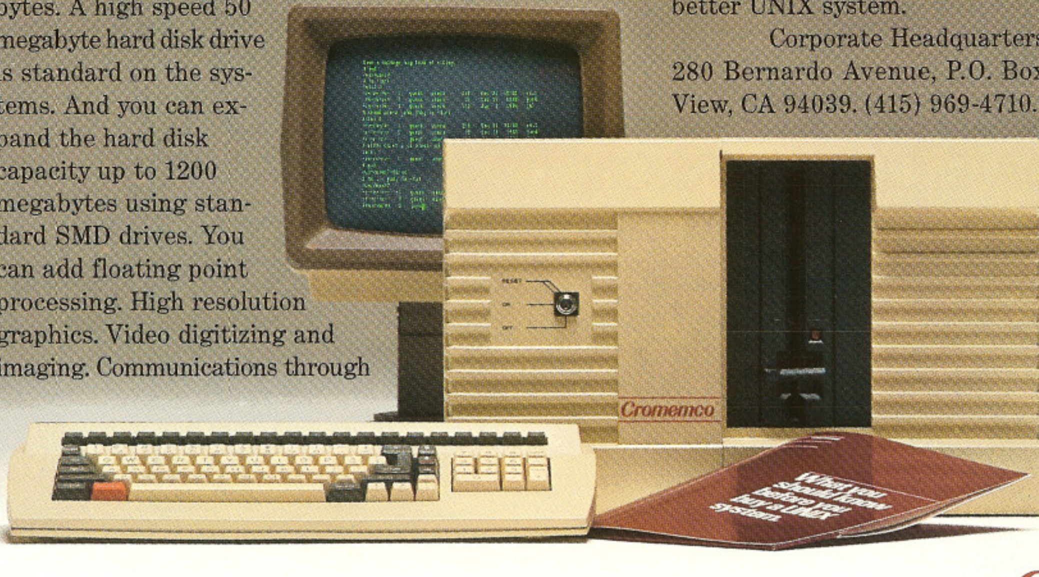
We also make our systems easier to use, because we install the operating system before we ship your computer. No complicated installation procedures. And the Berkeley enhancements give you the standard UNIX System V operating system, but with the added convenience of these widely acclaimed improvements.

Cromemco's System 100 and System 300 computers: designed to be the highest performance UNIX systems available anywhere.

Just call or visit one of our UNIX System V Official System Centers to see for yourself. They'll also give you a copy of our new publication, "What you should know before you buy a UNIX system." Or contact us directly.

We'll be glad to show you how to get a better UNIX system.

Corporate Headquarters: Cromemco, Inc., 280 Bernardo Avenue, P.O. Box 7400, Mountain View, CA 94039. (415) 969-4710. In Europe: Cromemco GmbH, 6236 Eschborn 1, Frankfurter Str. 33-35, P.O. 5267, Frankfurt Main, Germany.



UNIX is a trademark of Bell Laboratories.
IBM is a trademark of International Business Machines Corp.

Cromemco®